Interstage Application Development Cycle Manager V10.3

コマンドリファレンス

B1WD-2996-04Z0(00)
2012年12月

［高度な安全性が要求される用途への使用について］

本製品は、一般事務用、パーソナル用、家庭用、通常の産業等の一般的用途を想定して開発・設計・製造されているものであり、原子力施設における核反応制御、航空機自動飛行制御、航空交通管制、大量輸送システムにおける運行制御、生命維持のための医療用機器、兵器システムにおけるミサイル発射制御など、極めて高度な安全性が要求され、仮に当該安全性が確保されない場合、直接生命・身体に対する重大な危険性を伴う用途（以下「ハイセイフティ用途」という）に使用されるよう開発・設計・製造されたものではありません。お客さまは本製品を必要な安全性を確保する措置を施すことなくハイセイフティ用途に使用しないでください。また、お客さまがハイセイフティ用途に本製品を使用したことにより発生する、お客様または第三者からのいかなる請求または損害賠償に対しても富士通株式会社およびその関連会社は一切責任を負いかねます。

本マニュアルについて

本マニュアルでは、Interstage Application Development Cycle Manager（以降、ADM と呼びます）のコマンドラインインターフェースのインストールおよび使用方法について説明します。

本マニュアルの構成は、以下のとおりです。

タイトル		説明
1 章	はじめに	コマンドラインインターフェースの概要、インストールおよび使用方法、各コマンドのヘルプの表示方法について説明します。
2 章	リファレンス	コマンドラインインターフェースで使用できるすべてのコマンドのリファレンス情報について説明します。

本マニュアルの対象読者

本マニュアルは、ADM コマンドラインインターフェースを使用するすべてのユーザーを対象にしています。

以下の基礎知識が必要です。

- バッチファイルまたはシェルスクリプトの実行
- ADM の概念とコンポーネント

表記法

本マニュアルでは、以下の表記法を使用しています。

「Add」	メニューオプションなどの GUI 要素の名前は、括弧（「」）で囲んで記します。
`init`	コマンド名などのシステム名と、キーボードから入力するテキストは、Courier フォントで記します。
<変数>	値を入力する必要がある変数は、山括弧（< >）で囲んで記します。
	変数の値に特殊文字（空白、アンダースコア、ドットなど）が含まれる場合は、`Windows` では二重引用符（`" "`）で、`UNIX/Linux` では一重引用符（`' '`）で囲む必要があります。
	また、ワイルドカード文字（`'*'`または`'?'`）が含まれる場合は、`Windows` ではエスケープされた二重引用符（`¥" ¥"`）で、`Solaris/Linux` では一重引用符（`' '`）で囲む必要があります。
`[オプション]`	オプションのアイテム（オプションのコマンドパラメーターなど）は角括弧（`[ ]`）で囲んで記します。
`one | two`	いずれか 1 つを選択する項目は、縦棒（`|`）で区切って記します。
`{one | two}`	いずれか 1 つを選択する必須項目は、中括弧（`{ }`）で囲んで記します。

略語

本マニュアルでは、Interstage プロダクトとコンポーネントに以下の略語を使用します。

ADM	Interstage Application Development Cycle Manager
BPM	Interstage Business Process Manager
DMS	ドキュメント管理（Document Management System）コンポーネント
LCM	ライフサイクル管理（Lifecycle Management）コンポーネント（CHM、RLM、RQM）
SCM	ソフトウェア構成管理（Software Configuration Management）コンポーネント
SCCI	SCM のソースコードコントロールインターフェース（Source Code Control Interface）

関連ドキュメント

このマニュアルに加えて、ADM に関する以下のドキュメントを利用できます。

- 「解説書」：概要、主要機能、および基本コンセプトについて説明したマニュアルです。
- 「インストールガイド」：インストールについて説明したマニュアルです。
- 「アンインストールガイド」：アンインストールについて説明したマニュアルです。
- 「ライフサイクル管理（LCM） ユーザーズガイド」：ライフサイクル管理（LCM）コンポーネントをクライアントインターフェースおよび Web インターフェースから操作する方法を説明したオンラインマニュアルです。
- 「ソフトウェア構成管理（SCM） ユーザーズガイド」：ソフトウェア構成管理（SCM）コンポーネントをクライアントインターフェース、Web インターフェース、および Source Code Control Interface（SCCI）から操作する方法を説明したオンラインマニュアルです。
- 「ドキュメント管理（DMS） ユーザーズガイド」：ドキュメント管理（DMS）コンポーネントを、クライアントインターフェースおよび Web インターフェースから操作する方法を説明したオンラインマニュアルです。
- 「Subversion 連携ガイド」：ADM を Apache Subversion と統合する方法を説明したマニュアルです。
- 「管理者ガイド」：管理者向けの操作について説明したマニュアルです。
- 「カスタマイズガイド」：カスタマイズの方法を説明したマニュアルです。

目次

1 はじめに

この章では、ADM コマンドラインインターフェースの概要、インストール、使用方法および各コマンドのヘルプの表示方法について説明します。

1.1 コマンドラインインターフェースとは

コマンドラインインターフェースを使用すると、プログラミング言語を使用せずに、SCM と DMS のすべてのアクション、および LCM の基本的な機能を直接利用できます。

ADM コマンドラインインターフェースは、Windows 用のバッチファイルまたは UNIX/Linux 用のシェルスクリプトで構成されています。バッチファイルやシェルスクリプトは、コマンドおよびパラメーターとなる引数をあらかじめ組み合わせて定義しておいたものを使って呼び出すことができます。出力は、デフォルトの出力ストリームに返され、呼び出し側で調査、分析できます。

コマンドラインインターフェースには、以下の利点があります。

- 多くのサードパーティ製ツールがユーザー定義による外部プログラムの呼び出しをサポートしているので、SCM をサードパーティ製ツール（Microsoft Visual Studio など）に簡単に統合できます。
- DMS をサードパーティ製ツール（Microsoft Windows Explorer など）に簡単に統合できます。
- Windows の AT や、UNIX/Linux の cron などのスケジューリング機能を使用して、検索インデックス作成のような定期的なタスクを、ADM サーバ上で簡単に実行できます。
- Eclipse をインストールしなくても、ADM クライアントインターフェースに依存せずに、SCM、DMS、および LCM にアクセスできます。

コマンドラインインターフェースと並行してクライアントインターフェースを使う場合は、コマンドラインインターフェースで変更を行なった直後に、クライアントインターフェースに反映されるわけではないことに注意してください。それに応じてすべてのクライアントが更新されるには、時間がかかる場合があります。

ADM V10.3 では DMS は未サポートです。

1.2 接続モード

コマンドラインインターフェースのコマンドは、以下の接続モードで使用できます。

リモートモード：デフォルトの接続モードです。アクセス先のリポジトリが同じまたは異なる LAN（ローカルエリアネットワーク）にある場合のモードです。インターネットなどの WAN（ワイドエリアネットワーク）経由でリポジトリにアクセスできます。ADM には、Web サーバまたはアプリケーションサーバ経由でリポジトリにアクセスできる Web アプリケーションが付属しています。接続には、HTTP または HTTPS プロトコルを使用します。

ローカルモード：コマンドラインインターフェースとアクセス先のリポジトリが、同じ LAN 内のマシン上で動作する場合のモードです。リポジトリに直接アクセスできます。デフォルト接続モードがローカルモードに設定されている場合は、「-RC」オプションを指定して、コマンドラインインターフェースコールをリモートモードで実行できます。

SCM および DMS のコマンドは、ローカルモードでもリモートモードでも使用できます。

LCM のコマンドは、リモートモードでのみ使用できます。

1.3 インストールおよび使用方法

1.3.1 インストール

コマンドラインインターフェースのインストールの詳細は、「インストールガイド」で説明しています。要約すると、以下のどちらかの方法です。

- ADM サーバまたはクライアントのインストールプログラムを使用してコマンドラインインターフェースをシステムにインストールします。インストーラによって、利用可能なコマンドが異なります。例えば、esindex.bat コマンドはサーバインストーラを使用したときのみ利用できます。
- ADM インストールメディアの `commandline` フォルダーの内容を、サポート対象システムの任意のフォルダーにコピーします。

ADM コマンドラインインターフェースには、Java ランタイム環境（JRE）が必要です。詳細については、「インストールガイド」を参照してください。ADM クライアントおよびサーバセットアップは、ADM のインストールフォルダーの `jre` サブフォルダーに適切な JRE をインストールします。この JRE は、存在する場合に使われます。この JRE が存在しない場合は、`JAVA_HOME` 環境変数の設定で定義されている JRE が使用されます。

環境変数の設定

ADM コマンドラインインターフェースを実行するシステムには、以下の環境変数を任意に設定できます。

PATH

利便性のため、コマンドラインインターフェースのバッチファイルやシェルスクリプトを保存するフォルダーを、`PATH` 変数に追加できます。

ASM_VMARGS

この変数に、`-Duser.language=<id>`などの Java ランタイム環境のオプションを設定できます（以下を参照してください）。コマンドラインインターフェースは、これらのオプションを使用して、以下のように `java` 実行ファイルを呼び出します。

Windows：`%JAVA_HOME%¥bin¥java %ASM_VMARGS%`

UNIX/Linux：`$JAVA_HOME/bin/java $ASM_VMARGS`

コマンドラインインターフェースの言語

デフォルトでは、コマンドラインインターフェースでは、使用しているオペレーティングシステムの実行ユーザー用に設定されている言語が使用されます（Windows：「地域と言語のオプション」 UNIX/Linux：LANG 環境変数）。コマンドラインインターフェース経由で処理される ADM データに他の言語（日本語など）の文字が含まれる場合は、適切な言語設定を使用する必要があります。適切でない場合、データが文字化けすることがあります。

異なる言語を使用するには、Java システムプロパティの `user.language` を以下のように設定します。

```
-Duser.language=<id>
```

`<id>`は言語識別子です。英語の場合は `en`、日本語の場合は `ja` を設定します。

`user.language` プロパティは、`ASM_VMARGS` 環境変数として設定するのが最も良い方法です（上記を参照してください）。もう 1 つの方法として、コマンドラインインターフェースのバッチファイルまたはシェルスクリプトで、`java` 実行ファイルの起動オプションに追加することもできます。

1.3.2 コマンドの起動

コマンドラインインターフェースのエントリーポイントとして、以下のバッチファイルまたはシェルスクリプトを使用できます。

SCM および DMS：

- `escm.bat`（Windows）
- `escmant.bat`（Windows）：SCM コマンドを Apache ANT スクリプトから実行したい場合、`escm.bat` の代わりに、このバッチファイルを使用します。
- `escm.sh`（UNIX/Linux）

DMS：

- `edms.bat`（Windows）
- `edms.sh`（UNIX/Linux）

LCM：

- `elcm.bat`（Windows）
- `elcm.sh`（UNIX/Linux）

検索インデクサ：

- `esindex.bat`（Windows）
- `esindex.sh`（UNIX/Linux）

コマンドラインインターフェースのコマンドを実行するには、必要なパラメーターを使用して、適切なバッチファイルを呼び出します。バッチファイルまたはシェルスクリプトを 1 回呼び出すごとに、ADM コマンドを 1 つ実行できます。

また、以下のバッチファイル・シェルスクリプトを特定の目的で利用することができます。

- `elcmbug.bat/elcmbug.sh` は LCM の Bugzilla 連携機能のために利用することができます。詳細は「Interstage ADM 管理者ガイド」を参照してください。

バッチファイル・シェルスクリプトからのコマンド呼び出し

コマンドラインインターフェースで提供されているコマンドを、バッチファイルまたはシェルスクリプトから呼び出すことができます。

以下のことに注意してください。ADM で提供するコマンドのバッチファイルでは、echo をオフにしています（@echo off）。必要であれば、ADM のコマンドの呼び出し後に echo をオンに戻してください（@echo on）。

コマンド構文指定時の留意事項

コマンドを入力する場合には、以下にご留意ください。

- コマンドに引数が含まれている場合には、本マニュアルにある各コマンドの説明に従って、コマンド文字列の最後に引数を指定する必要があります。
- 同じコマンドオプションを複数回指定した場合には、最後のオプションだけが有効になります。
- コマンドラインインターフェースにおいては、ユーザーID、ユーザーグループ ID を指定する際には正規形式で入力してください。正規形式以外の形式で指定した場合、コマンドが失敗する可能性があります。正規形式については、「管理者ガイド」を参照してください。
- コマンドラインインターフェースにおいて変更した内容が、GUI 上に反映されるまでに数分間かかる可能性があります。
- コマンドオプションおよび引数は、コマンド構文の順番で指定するようにしてください。コマンド構文と異なる順番で入力した場合、エラーが発生する可能性があります。

1.3.3 SCMとDMSの共通コマンド

DMS は SCM をベースとしているので、DMS と SCM には多くの共通機能があります。したがって、一部のコマンドは SCM と DMS の両方で使用できます。

これらのコマンドのオプションの説明では、SCM 固有の用語が使用されています。これらのコマンドを DMS で使用する場合、以下の規則を適用してください。

- SCM プロジェクトまたはプロダクト：DMS ではライブラリまたはライブラリグループを指定します。
- SCM リリース：DMS と SCM を併用する場合は、対応する SCM リリースを指定します。SCM を使用せずに DMS を使用する場合は、リリースは常に `1.0` です。
- SCM コンフィグレーション：DMS と SCM を併用する場合は、対応する SCM コンフィグレーションを指定します。SCM を使用せずに DMS を使用する場合は、`MAIN` を指定してライブラリを指示するか、ライブラリエディションの名前を指定してエディションを指示します。

1.3.4 ヘルプの参照

オプションや引数を指定しないでバッチファイルまたはシェルスクリプトを呼び出すと、使用可能なすべてのコマンドのリストが表示されます。

特定のコマンドのオンラインヘルプ情報は、以下の形式で表示します。

```
<スタートスクリプト> --help <コマンド>
```

例

```
C:¥>escm --help list
```

1.3.5 SCMおよびDMSの認証情報の保存

escm コマンドまたは edms コマンドを使用してリポジトリにアクセスするときに必ず要求される認証情報（ユーザーID とパスワード）は、escm setauth コマンドを使用して保存できます。escm setauth コマンドは、利便性向上のために提供されています。このコマンドにより、認証情報は、ユーザーのホームフォルダー内のファイルに保存されます。このファイルが存在しない場合、escm コマンドまたは edms コマンドを使用してリポジトリにアクセスするごとに、認証情報を指定する必要があります。

もう 1 つの方法として、コマンドラインで呼び出すときに、ユーザー名とパスワードを直接指定することもできます。この場合、escm setauth によって提供される情報は無視されます。

```
escm <コマンド> ... [-eu <ユーザー名> -ep <パスワード>]
```

-eu <ユーザー名>
　リポジトリにアクセスするユーザーの名前を定義します。

-ep <パスワード>
　アクセスするユーザーのパスワードを定義します。

「–eu」パラメーターと「–ep」パラメーターは、escm と edms のバッチファイルで起動できるすべてのコマンドで使用できます。読みやすくするため、19頁の「リファレンス」のコマンドの説明にこれらのパラメーターは記載していません。他のバッチファイルによって呼び出されるコマンドは、escm setauth で保存された情報を参照しません。これらのコマンドには、対応するパラメーターを指定する必要があります。

1.3.6 LCMの接続情報の保存

elcm コマンドを使用してリポジトリにアクセスするときに必ず要求されるリポジトリ接続情報は、elcm setconnection コマンドを使用して保存できます。elcm setconnection コマンドは、利便性向上のために提供されています。このコマンドにより、リポジトリ接続情報は、ユーザーのホームフォルダー内のファイルに保存されます。このファイルが存在しない場合、elcm コマンドを使用してリポジトリにアクセスするごとに、接続情報を指定する必要があります。

もう 1 つの方法として、コマンドラインで呼び出すときに、リポジトリ接続情報を直接指定することもできます。この場合、elcm setconnection によって提供される接続情報は無視されます。

```
elcm <コマンド> ...  -d <リポジトリ名>
      -ru <リモート URL> [-rp <リモートプロキシ>] -eu <ユーザー名>
      -ep <パスワード>
```

-d <リポジトリ名>
　リポジトリの名前を定義します。

-ru <リモート URL>
　ADM Web アプリケーションが配置されている Web サーバまたはアプリケーションサーバにアクセスするリモートインターフェースの URL を定義します。

-rp <リモートプロキシ>
　ADM リモートインターフェースにアクセスするプロキシサーバの URL を定義します。

-eu <ユーザー名>
リポジトリにアクセスするユーザーの名前を定義します。

-ep <パスワード>
アクセスするユーザーのパスワードを定義します。

setconnection コマンドのオプションは、elcm のバッチファイルで起動できるすべてのコマンドに対して使用できます。読みやすくするため、147頁の「ライフサイクル管理」のコマンドの説明にこれらのパラメーターは記載していません。elcm setconnection コマンドまたは他の elcm コマンドを使用する場合、必須オプションを指定する必要があります。

1.3.7 エラー処理

コマンドラインインターフェースのバッチファイルは、コマンドが正常に実行されたかどうかを示す値を返します。以下のいずれかの値が返されます。

0：コマンドが正常に実行された。

1：コマンドの実行中に問題が発生した。

問題が発生した場合は、エラーメッセージが表示されます。エラーメッセージが表示されるのは、必要な変更をコマンドから実行できない場合だけです。例えば、すでにロックされているリソースをロックしても、メッセージは表示されません。

リポジトリのデータを追加または変更するアクションは、すべてトランザクション内で実行されます。したがって、アクションに失敗した場合は、すべての変更がロールバックされ、リポジトリは変更前と同じ状態になります。

ファイルシステム内のリポジトリ以外のファイルまたはフォルダーを修正するアクション（コンフィグレーションのエクスポートなど）はすべて、トランザクションでは保護されません。これらのアクションが何らかの問題によって中断された場合、ファイルシステムの一貫性が損なわれ、手作業での修復が必要になることがあります。

1.3.8 リソースパス規約

通常、ローカルリソースパスはすべて、ファイルシステム固有のパスセパレーターによって表示されます。Windows では「¥」、UNIX/Linux では「/」です。リポジトリ内のリソースを参照するパスはすべて、パスセパレーター「¥」を使用して表示されます。

ただし、コマンドオプションでリソースパスを指定する場合は、パスセパレーターとして「/」を使用します。

1.3.9 一時ファイルについて

ADM の一時ファイルは、コマンドラインインターフェースが使用する Java 仮想マシンに設定される一時フォルダーに保存されます。コマンドラインインターフェースをデフォルトのモードであるリモートモードで使用する場合は、一時フォルダーに、交換されるすべてのデータを保存できるだけの十分なディスクスペースが必要です。

一時フォルダーは、Java システムプロパティの java.io.tmpdir で指定します。このプロパティのデフォルト値は、オペレーティングシステムやそのバージョンに依存します。多くの場合、例えば Windows では多くの場合%UserProfile%¥Local Settings¥Temp、UNIX/Linux では/tmp です。java.io.tmpdir プロパティに対応する値を設定することにより、一時フォルダーを変更できます。例えば、以下のように指定します。

```
–Djava.io.tmpdir=D:¥Data¥IADM¥Temp
```

java.io.tmpdir プロパティは、ASM_VMARGS 環境変数で設定するのが最も良い方法です（11頁の「1.3.1 インストール」を参照してください）。もう 1 つの方法として、コマンドラインインターフェースのバッチファイルまたはシェルスクリプトで、Java 実行ファイルの起動オプションに追加することもできます。

1.3.10 ロギング

コマンドラインインターフェースの使用中に発生する問題は、ログファイルに書き出されます。デフォルトでは、以下のログ設定が使われます。

- ログ情報が、java.io.tmpdir Java システムプロパティで指定されている一時フォルダーに格納されているログファイルに書き込まれます（ログファイルのデフォルト名：${java.io.tmpdir}/asm_${user.name}.log)。>java.io.tmpdir プロパティの詳細については、15頁の「一時ファイルについて」を参照してください。
- ログレベルは WARN に設定されています。
- 10MB に設定されている最大ファイルサイズに達した場合には、ログファイルのバックアップが作成され、ログファイルは上書きされます。

log4j.properties コンフィグレーションファイルを編集してログ設定を変更できます。コンフィグレーションファイルは、インストールフォルダーの commandline サブフォルダーの中にあります。ログ設定の詳細については、「管理者ガイド」を参照してください。

1.4 ビルドスクリプトの例

以下に、Windows のビルドスクリプトの例を示します。このスクリプトは、SCM コンフィグレーションをチェックアウトし、ビルドスクリプトを呼び出して、ビルド結果を SCM リポジトリに保存します。最終的に、名前に日付/タイムスタンプを含むスナップショットが作成されます。

```
@ECHO OFF
REM SCM
REM ------------------------------------------------
REM Sample Windows script for an automatic daily build
REM ------------------------------------------------
ECHO SCM Sample Script
ECHO Started...

REM --- Define a working folder
SET SCMAB_TMPDIR=C:¥Test¥AutoBuild
RMDIR /S /Q %SCMAB_TMPDIR%
MKDIR %SCMAB_TMPDIR%

REM --- Define the configuration which should be built
SET SCMAB_SRV=estescmtstxp
SET SCMAB_REP=Sample
SET SCMAB_PRJ=Project B
SET SCMAB_REL=1.0
SET SCMAB_CFG=MAIN
SET SCMAB_USR=estescmtstxp¥Test
SET SCMAB_PWD=scm
SET SCMAB_PRJDIR=%SCMAB_TMPDIR%¥%SCMAB_PRJ%

REM --- Store authentication info in local temp file
call escm setauth -u %SCMAB_USR% -p %SCMAB_PWD%
IF ERRORLEVEL=1 goto errorend

REM --- Check-out latest resources to the local workspace
call escm checkout -s %SCMAB_SRV% -d %SCMAB_REP% -p "%SCMAB_PRJ%" -r
"%SCMAB_REL%" -c "%SCMAB_CFG%" -t "%SCMAB_PRJDIR%"
IF ERRORLEVEL=1 goto errorend

REM --- Call the project build file (generates build result)
call ant "%SCMAB_PRJDIR%¥build.xml
IF ERRORLEVEL=1 goto errorend

REM --- Commit the build results back to the repository
call escm commit -m "Auto-Build" "%SCMAB_PRJDIR%"
IF ERRORLEVEL=1 goto errorend

REM --- Create a snapshot of this build
call escm rtag -n "AUTOBUILD_%DATE%_%TIME%" -s %SCMAB_SRV% -
d %SCMAB_REP% -p "%SCMAB_PRJ%" -r "%SCMAB_REL%" -c "%SCMAB_CFG%"
IF ERRORLEVEL=1 goto errorend

REM --- Unregister the local project afterwards
cd "%SCMAB_PRJDIR%"
call escm disconnect –f
IF ERRORLEVEL=1 goto errorend
```

```
ECHO Ok.Finished build.
goto okend

:errorend
ECHO Aborted with error.

:okend

REM --- And remove authentication infos finally
call escm rmauth
```

1.5 IPv6 での運用

IPv6 での運用時における留意事項

IPv6 で運用する場合、以下の留意事項があります。

- 本製品は IPv4/IPv6 デュアルスタックのみサポートしています。IPv4 を無効にした場合の運用はサポートしておりません。
- 接続サーバを指定する場合、ホスト名で指定してください。IPv6 アドレスを使用することはできません。

IPv6 アドレスの指定方法

IPv6 アドレスを使用する場合、以下の hosts ファイルに IPv6 グローバルユニキャストアドレスとホスト名の宣言を追加します。宣言を追加したうえで、ホスト名を指定してください。

<Windows インストールフォルダー>¥system32¥drivers¥etc¥hosts

2 リファレンス

この章では、コマンドラインインターフェースの各種コマンドの機能および起動方法について説明します。

コマンドの適用範囲

各コマンド説明の見出しの後に、そのコマンドを適用できる ADM コンポーネントを略語で示します。

[SCM] SCM
[DMS] DMS
[LCM] LCM

コマンドを呼び出す一般的な形式は、以下のとおりです（SCM コマンドの例）：

```
escm [グローバルオプション] <コマンド> [オプション] [引数]
```

escm.bat バッチファイルを使用して、コマンド<コマンド>を呼び出します。

```
escm --help
```

escm.bat バッチファイルの一般的な使用方法に関するヘルプを表示します。

```
escm --help <コマンド>
```

escm.bat バッチファイルの、指定したコマンド<コマンド>の使用方法に関するヘルプを表示します。

2.1 認証およびリポジトリの接続

リポジトリにアクセスする SCM、DMS、または LCM の各コマンドには、有効なユーザーID とパスワードを指定する必要があります。この認証データは、以下のいずれかの方法で入力します。

- **コマンドオプション**

  各コマンドで、「-eu」オプションと「-ep」オプションを使用してユーザーID とパスワードを指定します。

- **対話モード**

  認証情報を指定しない場合、ユーザーID とパスワードの入力プロンプトがコマンドラインに表示されます。

  例

  ```
  C:¥>escm list -s estescmtstxp -d SCM213
  ```

- **認証ファイル**

  SCM、DMS、LCM の各コマンドの実行時に使用するユーザーID とパスワードを、ローカルファイルに永続的に保存できます。SCM と DMS では、setauth コマンドを使用して、必要な情報を含むローカルファイルを作成します。LCM では、setconnection コマンドを使用します。

認証プロセスは、アクセス先のリポジトリが置かれている ADM サーバの構成と、リポジトリの設定によって異なります。詳細については、「管理者ガイド」を参照してください。

2.1.1 setauth – 認証情報の保存 [SCM、DMS]

リポジトリにアクセスする SCM または DMS の各コマンドには、有効なユーザーID とパスワードを指定する必要があります。これらの認証データをユーザーのホームフォルダーに保存することによって、SCM または DMS のコマンドを簡単に呼び出すことができます。

引数を指定せずに実行した場合、対話形式でユーザーID とパスワードを入力します。

コマンド構文：

```
escm setauth [-u <ユーザーID> -p <パスワード>]
```

-u <ユーザーID>
: 認証に使用するユーザーの名前。

-p <パスワード>
: 認証に使用するユーザーのパスワード。

例

以降で入力するコマンドのユーザー名とパスワードを定義します。

```
C:¥>escm setauth -u estescmtstxp¥Test -p mypwd
```

2.1.2 rmauth – 認証ファイルの削除 [SCM、DMS]

セキュリティ上の理由から、ローカル環境にあるユーザー名やパスワードに関するすべての情報の削除が必要になることがあります。このコマンドを実行すると、setauth コマンドを使用して保存したすべての認証データが削除されます。

コマンド構文：

```
escm rmauth
```

例

すべての認証データを削除します。

```
C:¥>escm rmauth
```

2.1.3 setserver – サーバアクセスの定義 [SCM、DMS]

デフォルトでは、SCM または DMS の各コマンドは HTTP または HTTPS 接続を使用して、リモートモード（リモートインターフェース接続）で実行されます。ただし、そのためにはアプリケーションサーバに適切な iscm<バージョン>Web アプリケーションがインストールされ、起動されている必要があります（詳細については、「管理者ガイド」を参照してください）。接続するアプリケーションサーバを指定するために、このコマンドを使用する必要があります。これらの情報は、ユーザーのホームフォルダー内のファイルに保存されます。

ローカルモード（クライアント／サーバ接続）をデフォルトの接続モードとして使うこともできます。詳細は、「カスタマイズガイド」を参照してください。ローカルモードがデフォルトの接続モードとして定義されている場合は、コマンドラインインターフェースの escm コマンドまたは edms コマンドで「-RC」オプションを指定することで、リモートモードでコマンドは実行されます。

引数を指定しない場合、パラメーターの入力プロンプトが表示されます。

コマンド構文：

```
escm {setserver | ssv} [-h] [-u <ユーザーID> -p <パスワード>
      -n <プロキシ名または IP アドレス> -x <プロキシポート> -y <ポート>
      -w <web サーバ名> -c <コンテキスト>]
```

-h
: リモート接続で HTTPS プロトコルを有効にする場合（デフォルトのプロトコル：HTTP）。

-u <ユーザーID>
: プロキシ認証に使用するユーザーの名前（プロキシを使用し、認証が必要な場合のみ）。

-p <パスワード>
: プロキシ認証に使用するユーザーのパスワード（プロキシを使用し、認証が必要な場合のみ）。

-n <プロキシ名または IP アドレス>
: プロキシサーバの名前または IP アドレス（必要な場合のみ）。

-x <プロキシポート>
: プロキシサーバのポート番号（必要な場合のみ）。

-y <ポート>
: iscm<バージョン>Web アプリケーションに設定されたポート番号。

-w <web サーバ名>
: Web アプリケーションが配置されている Web サーバまたはアプリケーションサーバの名前または IP アドレス。

-c <コンテキスト>
: 対応する Web アプリケーションが待ち受けているコンテキストの名前。デフォルト

では、インストールされている SCM バージョンを使用して、コンテキストの名前が自動的に生成されます。/iscm<バージョン>（例 /iscm103）.

例

SCM リモートインターフェース接続を直接定義します。

```
C:¥>escm -RC setserver -w estenabler1 -y 80
```

SCM リモートインターフェース接続を対話的に定義します。

```
C:¥>escm setserver
```

2.1.4 setconnection – 接続情報の保存 [LCM]

LCM コマンドのリポジトリ接続情報を、ユーザーのホームフォルダー内のファイルに保存します。コマンドが正常に実行されると、ファイルのパス名が表示されます。

保存した接続情報は、以降で呼び出すすべての LCM コマンドに使用されます。ただし、コマンドに対応するオプションを明示的に指定する場合には、setconnection コマンドで保存したパラメーターは無視されます。

コマンド構文：

```
elcm {setconnection | scon} [-d <リポジトリ名> -ru <リモート URL> -rp
      <リモートプロキシ>
      -eu <ユーザー名> -ep <パスワード>]
```

-d <リポジトリ名>
LCM データが含まれているリポジトリの名前。

-ru <リモート URL>
リモートインターフェースの URL を、以下の形式で指定します。
http[s]://<サーバ>[:<ポート>]
<サーバ>は、Web アプリケーションが配置されているアプリケーションサーバです。
<ポート>は、サーバのアドレスを示すポート番号です。デフォルトでは、HTTP では 80、HTTPS では 443 が使用されます。

-rp <リモートプロキシ>
リモートインターフェースにアクセスするプロキシサーバの URL を、以下の形式で指定します。
<サーバ>[:<ポート>]
<サーバ>は、プロキシサーバの名前または IP アドレスです。<ポート>は、サーバのアドレスを示すポート番号です。

-eu <ユーザー名>
リポジトリに接続するユーザーの名前。指定しない場合、コマンドの実行時にユーザー名の入力プロンプトが表示されます。

-ep <パスワード>
ユーザーのパスワード。指定しない場合、コマンドの実行時にパスワードの入力プロンプトが表示されます。

例

リポジトリ「test」の接続情報を保存します。

```
elcm scon –d test -ru "http://server" –eu Guest
–ep Guest123
```

2.2 リポジトリの管理

リポジトリの作成および保守には、以下のコマンドを使用します。

2.2.1 connect – リポジトリの接続テスト [SCM、DMS]

指定したリポジトリへの接続およびリポジトリの状態を検査します。

コマンド構文：

```
escm connect -s <サーバ名> -d <リポジトリ名> [-po <ポート>]
```

-s <サーバ名>
　リポジトリが存在するサーバの名前。

-d <リポジトリ名>
　リポジトリの名前。

-po <ポート>
　リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

例

実行中ではないリポジトリに対して接続をテストします。

```
C:¥>escm connect -s estenabler1 -d SCM1
```

2.2.2 init – リポジトリの作成またはデータモデルの更新 [SCM、DMS]

SCM または DMS のリポジトリを作成します。指定したリポジトリがすでに存在する場合には、SCM または DMS で使用できるようにデータモデルが更新されます。

ADM の新リリースをインストールした後、リポジトリのデータモデルの変更またはエンハンスが必要になることがあります。データモデルの更新が必要な場合、ユーザーは新しいソフトウェアをインストールした後、そのリポジトリにアクセスできなくなります。ユーザーが古いリポジトリにアクセスしようとすると、エラーメッセージが表示されます。データモデルを更新するには、init コマンドを使用します。

コマンド構文：

```
escm init [{-model | -dn <リポジトリ表示名>}] -s <サーバ名> -d <リポジ
      トリ名> [-po <ポート>]
```

-model
　既存リポジトリのデータモデルを検査して、更新します。

-dn <リポジトリ表示名>
　リポジトリ表示名。このオプションで指定したリポジトリ表示名は、Interstage ADM クライアントインターフェース上でリポジトリを表す名前として使用されます。このオプションを省略した場合は、「-d」オプションで指定したリポジトリ名が使用されます。リポジトリ表示名には、「-d」オプションで指定したリポジトリ名と異なる名前を指定できます。リポジトリ表示名の長さは 255 バイトに制限されており、セミコロン（;）およびカンマ（,）以外の Unicode 文字を指定できます。

-s <サーバ名>
　リポジトリが存在する（またはリポジトリを保存する）サーバの名前。

-d <リポジトリ名>
リポジトリの名前。名前の長さは 14 バイトに制限されており、次の ASCII 7 ビット文字を指定できます: A-Z, a-z, 0-9, アンダースコア (_), ハイフン (-)

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

例

「Sample」という新しいリポジトリを作成します。

```
C:¥>escm init -s estescmtstxp -d Sample
```

2.2.3 start – リポジトリの起動 [SCM、DMS]

既存のリポジトリを起動します（リポジトリの管理者権限が必要です）。リポジトリは、使用する前に起動する必要があります。

コマンド構文：

```
escm {start | startDS} -s <サーバ名> -d <リポジトリ名> [-po <ポート>]
```

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
リポジトリの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

例

リポジトリ「Sample」を起動します。

```
C:¥>escm start -s estescmtstxp -d Sample
```

2.2.4 stop – リポジトリの停止 [SCM、DMS]

動作中のリポジトリを停止します（リポジトリの管理者権限が必要です）。特定の管理アクション（再構成など）を実行するには、事前にリポジトリを停止する必要があります。リポジトリを停止すると、ADM ユーザーはそのリポジトリにアクセスできなくなります。

コマンド構文：

```
escm {stop | stopDS} -s <サーバ名> -d <リポジトリ名> [-po <ポート>]
```

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
リポジトリの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、

9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

例

リポジトリ「Sample」を停止します。

```
C:¥>escm stop -s estescmtstxp -d Sample
```

2.2.5 drop – リポジトリの削除 [SCM、DMS]

特定のリポジトリを削除します（リポジトリの管理者権限が必要です）。このコマンドを実行するには、管理者権限が必要です。

注意：このアクション（削除）は元に戻すことができません。リポジトリのデータはすべて削除されます。

コマンド構文：

```
escm drop [-f] -s <サーバ名> -d <リポジトリ名> [-po <ポート>]
```

-f
　確認を要求せずに、リポジトリを削除します。

-s <サーバ名>
　リポジトリが存在するサーバの名前。

-d <リポジトリ名>
　リポジトリの名前。

-po <ポート>
　リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

例

リポジトリ「Sample」を削除します。

```
C:¥>escm drop -s estescmtstxp -d Sample
```

2.2.6 sett – リポジトリ全般の設定の定義 [SCM、DMS]

リポジトリ全般の設定を定義します。これらの値を変更できるのは、リポジトリ管理者だけです。

コマンド構文：

```
escm sett -s <サーバ名> -d <リポジトリ名> [-po <ポート>
      -exh <ファイル名> -exhjar <ファイル名> -ds <on/off>
      -ral <on/off> -pcr <on/off> -pcgid <グループ ID（複数可）>
      -dbgid <グループ ID（複数可）-sn <サーバ名>
      -delexhjar <拡張出口アーカイブ>] [-getexh <ファイル名>]
      [-getexhjar <拡張出口アーカイブ>
      [-t <ファイル名>] ]
```

-s <サーバ名>
　リポジトリが存在するサーバの名前。

-d <リポジトリ名>
　リポジトリの名前。

-po <ポート>
　リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-exh <ファイル名>
　このオプションは今版では利用できません。

-exhjar <ファイル名>
　このオプションは今版では利用できません。

-ds <on/off>
　「on」を指定すると、ファイルバージョンの差分管理が有効になります。「off」を指定すると、無効になります。デフォルトでは「off」になります。

-ral <on/off>
　「on」を指定すると、読み取りアクセスのロギングが有効になります。「off」を指定すると、無効になります。デフォルトでは「off」になります。

-pcr <on/off>
　「on」を指定すると、新規プロジェクトの作成がリポジトリ管理者だけに制限されます。「off」を指定した場合は、どのユーザーでもプロジェクトを作成できます。デフォルトでは「off」になります。これはプロジェクト作成のためのユーザーグループが指定されていない場合にのみ有効です（「-pcgid」オプションを参照）。

-pcgid <グループ ID（複数可）>
　リポジトリサーバが認識している最大 8 のユーザーグループを、セミコロンで区切って入力します。指定したユーザーグループのメンバーにプロジェクトの作成が許可されます。この設定を変更できるのは、（プロジェクトの作成が常に許可されている）リポジトリ管理者だけです。値に「null」を指定すると、グループの定義が削除されます。

-dbgid <グループ ID（複数可）>
　リポジトリサーバが認識している最大 8 のユーザーグループを、セミコロンで区切って入力します。指定したユーザーグループのメンバーにリポジトリへの接続が許可されます。この設定を変更できるのは、（リポジトリへの接続が常に許可されている）リポジトリ管理者だけです。値に「null」を指定すると、グループの定義が削除され、すべての既知ユーザーにリポジトリへの接続が許可されます（デフォルト）。

-sn <サーバ名>
　SCM または DMS リポジトリへの接続に常に使用される統一されたサーバ名。すべてのユーザーが、この名前を使って、サーバにアクセスできる必要があります。ユーザーが、サーバ名として短縮名や完全修飾名などの様々なアドレスを使ってサーバに接続している LAN 環境では、統一されたサーバ名を定義することを推奨します。LCM から SCM アイテムへのリンクを作成したり、検索インデックスを作成/更新したり、検索クエリを実行したりする際に、この統一されたサーバ名が使われます。リポジトリ作成時、統一されたサーバ名として、作成時に使われた接続サーバ名がプリセットされます。

-delexhjar <拡張出口アーカイブ>
　このオプションは今版では利用できません。

-getexh <ファイル名>
　このオプションは今版では利用できません。

-getexhjar <拡張出口アーカイブ> [-t <ファイル名>]
　このオプションは今版では利用できません。

例

ユーザーグループ「estescmtstxp¥escm」のすべてのメンバーにプロジェクトの作成を許可します。

```
C:¥>escm sett -s estescmtstxp -d Sample2 -pcgid estescmtstxp¥escm
```

2.2.7 chmansett – 変更管理の設定の定義 [SCM, DMS]

SCM、DMS のリポジトリに対して、変更管理の設定を定義します。SCM、DMS で変更管理を有効にするためには、関連するタスクを持つ LCM リポジトリへの接続が定義されている必要があります。

変更管理への自動反映

LCM のリリースとタスクに自動的に変更を反映することができます。この場合、SCM、DMS で行われたすべてのオペレーションが自動的に LCM に反映されます。例えば、SCM でリリースを作成すると、対応する同名のリリースが自動的に LCM に作られます。

このように LCM を使用できるようにするには、変更管理への自動反映を有効にする必要があります。

注意: 変更管理への自動反映を使用する場合、SCM プロジェクトまたは DMS ライブラリと LCM アプリケーションとの間で 1 対 1 の関係となるように SCM または DMS だけを使用して変更を管理していることを前提にしています。LCM で直接変更した場合には、予期せぬ結果やエラーを引き起こす可能性があることに注意してください。例えば、SCM でプロジェクトを削除した場合、対応する LCM アプリケーションに複数のプロジェクトが関連付けてられている場合でも、そのアプリケーションは LCM で削除されます。変更管理の自動反映に切り替える場合は、すでに定義済みの関連が、1 対 1 の関係であることを確認してください。

コマンド構文:

```
escm chmansett -s <サーバ名> -d <リポジトリ名>[-po <ポート>
      -e <0/1(有効化)> -m <0/1(有効化)> -cs <サーバ>
      -cd <リポジトリ> -cpo <ポート> -cru <リモート URL>]
```

-s <サーバ名>
SCM または DMS のリポジトリが存在するサーバ名

-d <リポジトリ名>
SCM または DMS のリポジトリ名

-po <ポート>
SCM または DMS リポジトリが存在するサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-e <0/1(有効化)>
値が"1"のとき、変更管理が有効。値が"0"のときは無効。

-m <0/1(有効化)>
値が"1"のとき、変更管理への自動反映が有効。値が"0"のときは無効。デフォルトの値は"0"となります。

-cs <サーバ>
LCM のリポジトリが存在するサーバ名

-cd <リポジトリ>
関連するタスクが保存されている LCM リポジトリの名前

-cpo <ポート>
LCM リポジトリが存在するサーバにアクセスするためのポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-cru <リモート URL>
LCM 用のリモートインターフェースの URL。以下のフォーマットを使用してくださ

い。
http[s]://<サーバ>[:<ポート>]
<サーバ> は、iasmcore<バージョン> Web アプリケーションを配備したアプリケーションサーバ。
<ポート> は、アプリケーションサーバにアクセスするためのポート（デフォルトでは HTTP では 80、HTTPS では 443 が使用されます）。

例

リポジトリ”Sample”に対して、変更管理の設定を定義します。

```
C:¥>chmansett -s estescmtstxp -d Sample -po 9701
```

2.2.8 mailsett – メール設定の定義 [SCM、DMS]

コミット通知機能などに使用される、SCM または DMS のメール通知機能の設定を定義します。

コマンド構文：

```
escm mailsett -s <サーバ名> -d <リポジトリ名> [-po <ポート>
      -e <0/1(有効化)> -h <SMTP サーバ名> -m <送信者> -u <ユーザーID>
      -z <ユーザーパスワード>]
```

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
リポジトリの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-e <0/1(有効化)>
「1」を指定するとメール通知が有効になり、「0」(デフォルト)を指定すると無効になります。

-h <SMTP サーバ名>
SMTP サーバの名前または IP アドレス。メール通知機能が有効な場合、サーバ名は必ず指定されていなければなりません。次の文字を指定できます: A-Z, a-z, 0-9, コロン (:), ドット (.), アンダースコア (_), プラス (+), およびハイフン (-)

-m <送信者>
メールの送信者として使われるユーザーID またはメールアドレス。メール通知機能が有効な場合、送信者は必ず指定されていなければなりません。次の文字を指定できます: A-Z, a-z, 0-9,ドット (.), アンダースコア (_), プラス (+), ハイフン (-), およびアットマーク (@)

-u <ユーザーID>
SMTP サーバへログインする際のユーザーID（SMTP サーバがメール送信時に認証を必要とする場合のみ）

-z <ユーザーパスワード>
SMTP サーバへログインする際のユーザーパスワード（SMTP サーバがメール送信時に認証を必要とする場合のみ）。

例

リポジトリのメール設定を定義します（SMTP サーバが認証を必要としない場合）。

```
C:¥>escm mailsett -s estescmtstxp -d Sample -e 1 -h SMTPSRV1
-m "SCM Mailer" -z ""
```

メールシステムを無効にします。

```
C:¥>escm mailsett -s estescmtstxp -d Sample -e 0
リポジトリ "Sample" に接続中。しばらくお待ちください。...接続完了.
```

2.2.9 searchsett – 検索設定の定義 [SCM、DMS]

リポジトリでの SCM と DMS の全文検索の設定を定義します。前提条件として、Interstage ADM 検索サーバをインストールしたマシン上に、リポジトリ用の全文検索インデックスが作成されている必要があります。

コマンド構文：

```
escm searchsett -s <サーバ名> -d <リポジトリ名> [-po <ポート>
      -e <0/1(有効化)> -u <検索サーバ> -z <インデックスパス>]
```

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
リポジトリの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「カスタマイズガイド」を参照してください。

-e <0/1(有効化)>
「1」を指定すると検索機能が有効になり、「0」(デフォルト)を指定すると無効になります。

-u <検索サーバ>
iscm<バージョン>Web アプリケーションが配備され、対象のリポジトリの検索インデックスが格納されている検索サーバの URL。URL は以下のフォーマットで指定します。
http[s]://<サーバ名>[:<ポート>]
<サーバ名>は Interstage ADM Web アプリケーションが配備されている Web サーバまたはアプリケーションサーバの名前または IP アドレスです。<ポート>は Web サーバまたはアプリケーションサーバにアクセスするポート番号です。ポート番号は、デフォルトでは HTTP の場合は 80、HTTPS では 443 が使用されます。

-z <インデックスパス>
検索サーバ上の、リポジトリの検索インデックスが存在するフォルダーの絶対パス名。インデックスパスの最大長は 255 バイトとなります。

例

リポジトリの検索設定を定義します。

```
C:¥>escm searchsett -s estescmtstxp -d Sample -e 1 -u http://searchsrv1
-z "D:¥Data¥ADM_Search"
```

リポジトリの全文検索を無効にします。

```
C:¥>escm searchsett -s estescmtstxp -d Sample -e 0
```

2.2.10 usermail – メールアドレスの表示または定義 [SCM、DMS]

指定したユーザーのメールアドレスを表示または定義します。

有効なメールアドレスが指定されていない場合、ユーザーはメール通知を受けることができません。。

コマンド構文：

```
escm usermail -s <サーバ名> -d <リポジトリ名> -u <ユーザー名>
      [-po <ポート> -m <メールアドレス>]
```

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
リポジトリの名前。

-u <ユーザー名>
ユーザーの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-m <メールアドレス>
ユーザーの新しいメールアドレス。「-m」オプションを省略すると、指定したユーザーの既存のメールアドレスが表示されます。

例

ユーザーの既存メールアドレスを表示します。

```
C:¥>escm usermail -s estescmtstxp -d Sample -u estescmtstxp¥user1
```

2.2.11 expdelprjlogs – 削除したプロジェクトのログファイルのエクスポート [SCM]

指定したリポジトリで、削除されたすべての SCM プロジェクトのログファイルを検索してエクスポートします。

ログファイルは、XML 形式でエクスポートされます。エクスポートされる XML ファイルの名前は、SCMProject_<プロジェクトキー>.xml です。<プロジェクトキー>は、削除された SCM プロジェクトの内部 ID です。

「**ログファイルの再構成**」機能を使用してリポジトリからログファイルを永久に削除すると、削除されたログファイルは以降で検索できなくなります。

コマンド構文：

```
escm expdelprjlogs -s <サーバ名> -d <リポジトリ名> [-po <ポート>]
      <フォルダー名>
```

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
処理するリポジトリの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

<フォルダー名>
XML ファイルのエクスポート先となるフォルダーのパス名。生成される XML ファイルと同じ名前のファイルが存在する場合、上書きされます。

例

リポジトリ「Sample」から、削除されたプロファイルのログファイルをフォルダー「C:¥tmp¥logfiles」にエクスポートします。

```
C:¥>escm expdelprjlogs -s estescmtstxp -d Sample C:¥tmp¥logfiles
```

2.3 プロジェクトの管理

SCM プロジェクトおよび DMS ライブラリの作成および保守には、以下のコマンドを使用します。

2.3.1 creaprj – プロジェクトの作成 [SCM]

リポジトリに SCM プロジェクトを作成します。新規プロジェクトには、初期リリース、作業コンフィグレーション、および基本コンフィグレーションが含まれます。作業コンフィグレーションと基本コンフィグレーションは、空の状態です。

変更管理が有効で、SCM リポジトリとの関連付けが設定されていれば、新規プロジェクトに関連するアプリケーションとリリースが LCM で作成されます。新規 LCM アプリケーションは、SCM プロジェクトと同じ名前で作成されます。LCM アプリケーションでは、必要に応じてコマンドを実行したユーザーに 1 つ以上のユーザーロールが割り当てられます。これは、ユーザーに割り当てられている SCM ユーザーロールに含まれるすべての関連するオーソリティを LCM オーソリティにマップすることで実施します。関連するオーソリティは、SCM アクションを LCM に反映させるために必要となるオーソリティです。さらに、LCM リポジトリでデフォルトロールが定義されていない場合は、デフォルトロールが、SCM のデフォルトロールで指定されているすべての関連するオーソリティを含むロールに設定されます。

SCM または DMS と LCM 間の関連するオーソリティのマッピングの詳細については、51 頁の「ロールとオーソリティ」を参照してください。

プロジェクト名の制限

各種のクライアントオペレーティングシステムを適正にサポートするために、プロジェクトの名前には以下の文字と同様に、すべての制御文字を使用できません。
`* : ¥ / ? < > | "`
プロジェクト名にこれらの文字が含まれていると、プロジェクトの作成に失敗します。

コマンド構文：

```
escm creaprj [-h -nolcm] -s <サーバ名> -d <リポジトリ名>
      -n <プロジェクト名> -g <グループ ID（複数可）> [-po <ポート>]
```

`-h`
非表示プロジェクトを作成します。非表示プロジェクトは、標準プロジェクトのリストには含まれません。クライアントインターフェース（Eclipse）では表示され、プロダクトとして処理されます。

`-nolcm`
LCM への接続を無効にします。この場合、LCM のアプリケーションとリリースは作成されません。「変更管理への自動反映」がリポジトリで有効になっている場合は（`chmansett` コマンドを参照）、このオプションで接続を無効にできません。

`-s <サーバ名>`
リポジトリが存在するサーバの名前。

`-d <リポジトリ名>`
新規プロジェクトを保存するリポジトリの名前。

`-n <プロジェクト名>`
新規プロジェクトの名前。

`-g <グループ ID（複数可）>`
リポジトリサーバが認識している最大 8 のユーザーグループをセミコロンで区切っ

て入力します。指定したユーザーグループのメンバーは、新規プロジェクトで作業ができます。

`-po <ポート>`
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、`9500` です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

例

2 つのアクセスグループを指定した「Project A」というプロジェクトを作成します。

```
C:¥>escm creaprj -s estescmtstxp -d Sample -n "Project A"
```

2.3.2 share – ローカルプロジェクトの共有 [SCM]

ローカルワークスペースの既存プロジェクトを、指定したリポジトリに SCM プロジェクトとして登録し、デフォルトのリリース、作業コンフィグレーション、基本コンフィグレーションを作成します。作業コンフィグレーションと基本コンフィグレーションは、空の状態です。

作業コンフィグレーションには、主要な開発ブランチのリソースとしてすべてのフォルダーとファイルが保存されます。リソースを追加するには、`add` コマンドまたは `commit` コマンドを使用します。

変更管理が有効で、SCM リポジトリとの関連付けが設定されていれば、新規プロジェクトに関連するアプリケーションとリリースが LCM で作成されます。新規 LCM アプリケーションは、SCM プロジェクトと同じ名前で作成されます。LCM アプリケーションでは、必要に応じてコマンドを実行したユーザーに 1 つ以上のユーザーロールが割り当てられます。これは、ユーザーに割り当てられている SCM ユーザーロールに含まれるすべての関連するオーソリティを LCM オーソリティにマップすることで実施します。関連するオーソリティは、SCM アクションを LCM に反映させるために必要となるオーソリティです。さらに、LCM リポジトリでデフォルトロールが定義されていない場合は、デフォルトロールが、SCM のデフォルトロールで指定されているすべての関連するオーソリティを含むロールに設定されます。

SCM または DMS と LCM 間の関連するオーソリティのマッピングの詳細については、51 頁の「ロールとオーソリティ」を参照してください。

プロジェクトを共有すると、プロジェクトチームのメンバーはリソースを各自のローカルワークスペースにチェックアウトできます。

コマンド構文：

```
escm {share | sh} [-nolcm] -s <サーバ名> -d <リポジトリ名>
      -n <プロジェクト名> -g <グループ ID（複数可）> [-po <ポート>]
      <ルートフォルダー名>
```

`-nolcm`
LCM への接続を無効にします。この場合、LCM のアプリケーションとリリースは作成されません。「変更管理への自動反映」がリポジトリで有効になっている場合は（`chmansett` コマンドを参照）、このオプションで接続を無効にできません。

`-s <サーバ名>`
リポジトリが存在するサーバの名前。

`-d <リポジトリ名>`
新規プロジェクトを保存するリポジトリの名前。

-n <プロジェクト名>
新規プロジェクトの名前。

-g <グループ ID（複数可）>
リポジトリサーバが認識している最大 8 のユーザーグループをセミコロンで区切って入力します。指定したユーザーグループのメンバーは、新規プロジェクトで作業ができます。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態ではデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

<ルートフォルダー名>
共有するローカルプロジェクトのルートフォルダーのパス名。

例

「projects¥Demo」に存在するローカルプロジェクトを共有します。

```
C:¥test>escm share -s estescmtstxp -d Sample -n "Project C"
-g estescmtstxp¥escm;estescmtstxp¥Administrators projects¥Demo
```

2.3.3 updshare – ローカルプロジェクトの共有の更新 [SCM]

SCM リポジトリまたは作業コンフィグレーションへのローカルプロジェクトの割当てを変更します。既存の割当ては削除されます。リポジトリを移動した場合やリポジトリ名を変更した場合、このコマンドを使用します。

コマンド構文：

```
escm updshare -s <サーバ名> -d <リポジトリ名> -p <プロジェクト名>
      -r <リリース名> -c <コンフィグレーション名> [-po <ポート>]
      <フォルダー名>
```

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
指定したローカルプロジェクトに割り当てるリポジトリの名前。

-p <プロジェクト名>
指定したローカルプロジェクトに割り当てるプロジェクトの名前。

-r <リリース名>
指定したローカルプロジェクトに割り当てるリリースの名前。

-c <コンフィグレーション名>
指定したローカルプロジェクトに割り当てるコンフィグレーションの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態ではデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

<フォルダー名>
割り当てを変更するローカルプロジェクトのフォルダーのパス名。

例

ローカルプロジェクト「PrjA」の割り当てを、指定したコンフィグレーションに変更します。

```
C:¥>escm updshare -s estenabler1 -d SCM1 -p Demo -r 2.00 -c MAIN
"/Dev/PrjA"
```

2.3.4 disconnect – SCMからのローカルプロジェクトの解放 [SCM]

注意: このコマンドは、＜フォルダー名＞の引数を指定しない場合、現在の作業フォルダーに適用されます。

リポジトリ内のプロジェクトとローカルワークスペースのプロジェクト間の接続を削除します。このアクションを実行すると、ローカルリソースは SCM によって管理されなくなります。

SCM メタ情報を含むすべてのファイルが、ローカルワークスペースのプロジェクトのフォルダー構造全体から削除されます。リポジトリは変更されません。

コマンド構文：

```
escm {disconnect | dcon} [-r -f] [<フォルダー名>]
```

-r
: プロジェクトのローカルフォルダー構造を削除します。

-f
: 確認を要求せずに、アクションを実行します。

<フォルダー名>
: ローカルプロジェクトに属している任意のフォルダーのパス名。指定しない場合、現在の作業フォルダーが使用されます。

例

ローカルリソースを保持したままで、バージョン管理からローカルプロジェクトを解放します。

```
C:¥test¥co2¥Demo>escm disconnect
```

2.3.5 updproj – プロジェクトプロパティの更新 [SCM、DMS]

SCM プロジェクトまたは DMS ライブラリのプロパティを変更します。

コマンド構文：

```
escm updproj -s <サーバ名> -d <リポジトリ名> -p <プロジェクト名>
      [-po <ポート> -g <グループ ID（複数可）> -w <on/off>
      -z <on/off> -cv <on/off> -y <on/off> -k <on/off> -kenc <文字
      コード> -prp <プロパティ;...>]
```

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
プロジェクトまたはライブラリが含まれているリポジトリの名前。

-p <プロジェクト名>
プロジェクトまたはライブラリの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-g <グループ ID（複数可）>
リポジトリサーバが認識している最大 8 のユーザーグループをセミコロンで区切って入力します。指定したユーザーグループのメンバーは、プロジェクトまたはライブラリで作業ができます。

-w <on/off>
「on」を指定すると、リポジトリ管理者のロールチェックが有効になります。「off」（デフォルト）を指定すると無効になります。無効にした場合、リポジトリ管理者にすべてのオーソリティが自動的に割り当てられます。

-z <on/off>

「on」を指定すると、厳密な変更制御が有効になります。「off」を指定すると、無効になります。有効にした場合、ユーザーがプロジェクトまたはライブラリにリソースをコミットまたは追加するときに、最低 1 つのタスクを指定する必要があります。タスクが割り当てられていないユーザーは、コミットおよび追加をすることができません。

-cv <on/off>

「on」を指定すると、コミット検証およびチェックイン検証が有効になります。「off」を指定すると、無効になります。有効にした場合、ユーザーがプロジェクトまたはライブラリにリソースをコミットまたはチェックインするときに、タスクでマークされたドキュメントまたはフォルダーだけしかコミットまたはチェックインできなくなります。コミット検証およびチェックイン検証を有効にした場合、自動的に厳密な変更制御も有効になります。

-y <on/off>
「on」を指定すると、ローカルリソースの監視が強制的に有効になります。「off」を指定すると無効になります。有効にした場合、チェックアウトしたリソースのすべての編集作業がリポジトリに記録されます。無効にした場合、共有プロジェクトにローカルリソースの監視を使用するかどうかをユーザーが決定できます。

-k <on/off>
「on」を指定すると、コミット時のキーワード置換が有効になります。「off」を指定すると、無効になります。キーワード置換が無効になっている場合、ソースファイルは全く変更無しにリポジトリに保存されます。

-kenc <文字コード>
キーワード置換のための文字コードを定義します。指定しない場合、実行環境にて使われているデフォルトの文字コードが利用されます。実行環境（現在利用している JRE）の文字コードは、デフォルトでは OS により定義される言語となります。

文字コードの例: Shift_JIS, EUC_JP, UTF-8

-k が指定されていない場合、-kenc は無視されます。

-prp <プロパティ>
このオプションは今版では利用できません。

例

「Project B」のアクセスグループとして、ユーザーグループ「users1」を定義します。

```
C:¥>escm updproj -s estescmtstxp -d Sample -p "Project B" –g users1
```

「Project B」のデータストア管理者のロールチェックを有効にします。

```
C:¥>escm updproj -s estescmtstxp -d Sample -p "Project B" –w on
```

2.3.6 renproj – プロジェクトの名前変更 [SCM]

指定した SCM プロジェクトの名前を変更します。変更管理が有効で、SCM リポジトリとの関連付けが設定されている場合、LCM の関連アプリケーションの名前も変更されます。

プロジェクト名の制限

各種のクライアントオペレーティングシステムを適正にサポートするために、プロジェクトの名前には以下の文字と同様に、すべての制御文字を使用できません。
* : ¥ / ? < > | "
プロジェクト名にこれらの文字が含まれていると、プロジェクトの名前変更に失敗します。

コマンド構文：

```
escm renproj [-nolcm] -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> [-po <ポート>] <新しいプロジェクト名>
```

-nolcm
LCM への接続を無効にします。この場合、LCM の関連するプロジェクトリンクは更新されません。「変更管理への自動反映」がリポジトリで有効になっている場合は（chmansett コマンドを参照）、このオプションで接続を無効にできません。

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
プロジェクトが含まれているリポジトリの名前。

-p <プロジェクト名>
プロジェクトの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

<新しいプロジェクト名>
　　変更後のプロジェクトの名前。

例

「Project A」の名前を「Project B」に変更します。

```
C:¥>escm renproj -s estescmtstxp -d Sample -p "Project A" "Project B"
```

2.3.7 delproj – プロジェクトの削除 [SCM]

既存の SCM プロジェクトを削除します。変更管理が有効で、SCM リポジトリとの関連付けが設定されている場合、プロジェクトに関連するアプリケーションとリリースも LCM で削除できます。

注意：このアクション（削除）は元に戻すことができません。プロジェクトのデータはすべて削除されます。

削除の実行中にキャンセルすると、プロジェクトのデータの一部がリポジトリ内に残ります。この場合、再度このコマンドを実行し、プロジェクトを完全に削除してください。

コマンド構文：

```
escm delproj [-f -nolcm] -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> [-po <ポート>]
```

-f
　　確認を要求せずに、プロジェクトを削除します。

-nolcm
　　LCM への接続を無効にします。この場合、LCM の関連するアプリケーションとリリースは削除されません。「変更管理への自動反映」がリポジトリで有効になっている場合は（chmansett コマンドを参照）、このオプションで接続を無効にできません。

-s <サーバ名>
　　リポジトリが存在するサーバの名前。

-d <リポジトリ名>
　　プロジェクトが含まれているリポジトリの名前。

-p <プロジェクト名>
　　削除するプロジェクトの名前。

-po <ポート>
　　リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

例

既存プロジェクトを削除します。

```
C:¥>escm delproj -s estescmtstxp -d Sample -p "Project C"
```

2.3.8 sharesett – プロジェクトまたはライブラリの設定の共有 [SCM、DMS]

指定したプロジェクトまたはライブラリで、特定の SCM プロジェクトまたは DMS ライブラリのすべての設定を共有します。

コマンド構文：

```
escm sharesett [-remove] -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> [-po <ポート>] [<共有プロジェクト名>]
```

-remove
　プロジェクト設定の共有を取り消します。

-s <サーバ名>
　リポジトリが存在するサーバの名前。

-d <リポジトリ名>
　プロジェクトまたはライブラリが含まれているリポジトリの名前。

-p <プロジェクト名>
　プロジェクトまたはライブラリの名前。

-po <ポート>
　リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

<共有プロジェクト名>
　設定を共有するプロジェクトまたはライブラリの名前。

例

「Project A」で「Project Master」の設定を共有します。

```
C:¥>escm sharesett -s estescmtstxp -d Sample -p "Project A" "Project
Master"
```

「Project A」での設定の共有を取り消します。

```
C:¥>escm sharesett -s estescmtstxp -d Sample -p "Project A" –remove
```

2.3.9 exportkeywords – キーワード定義のエクスポート [SCM]

SCM プロジェクトに定義されているキーワード定義を XML ファイルにエクスポートします。この XML ファイルは新しいプロジェクトにインポートしたり、テンプレートとして利用したりすることができます。

コマンド構文：

```
escm {exportkeywords | exk} -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> [-po <ポート> -t <エクスポート先のファイル名>]
```

-s <サーバ名>
　リポジトリが存在するサーバの名前。

-d <リポジトリ名>
　プロジェクトが含まれているリポジトリの名前。

-p <プロジェクト名>
　プロジェクトの名前。

-po <ポート>
　リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-t <エクスポート先のファイル名>
キーワード定義をエクスポートする先となる XML ファイルのパスおよびファイル名。本オプションを指定しなかった場合、データは「<プロジェクト名>_keywords.xml」という名前のファイルにエクスポートされ、現在の作業フォルダーに保存されます（<プロジェクト名>は、「-p」オプションで指定したプロジェクト名です）。指定したファイルがすでに存在する場合、上書きされます。

例

「ProjectA」のキーワード定義をエクスポートします。

```
C:¥>escm exportkeywords -s estescmtstxp -d Sample -p "Project A" -t
C:¥Tmp¥keywords.xml
```

2.3.10 importkeywords – キーワード定義のインポート [SCM]

SCM プロジェクトにキーワード定義をインポートします。インポートされたキーワードで現在定義されているキーワードのリストは置き換えられます。

コマンド構文：

```
escm {importkeywords | imk} -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -t <インポート元のファイル名> [-po <ポート>]
```

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
プロジェクトが含まれているリポジトリの名前。

-p <プロジェクト名>
プロジェクトの名前。

-t <インポート元のファイル名>
インポートされる XML ファイルのパス名

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

例

「ProjectA」に"C:¥tmp¥keywords.xml" ファイルをインポートします。

```
C:¥>escm importkeywords -s estescmtstxp -d Sample -p "Project A"
-t C:¥Tmp¥keywords.xml
```

2.3.11 newstate – ステートの作成 [SCM]

指定した SCM プロジェクトのコンフィグレーションステートを作成します。コンフィグレーションで各ステートを使用できるようにするには、各ステートをプロジェクト用に定義する必要があります。

ソフトウェア開発プロセスはいずれも、企業固有のライフサイクルに基づいています。つまり、各コンフィグレーションは、ある時点での 1 つの定義されたステートになります。作業コンフィグレーションは、通常、「in development（開発中）」のステートですが、保護されたコンフィグレーションは、「quality assurance（品質保証）」または「pre-production test（試作テスト）」のステートになります。また、日常のビルドなどの未使用コンフィグレーションは、最終ステートの「archived（アーカイブ済み）」になり、リ

リース済みコンフィグレーションは、「released（リリース済み）」または「delivered（提供済み）」のステートになります。

ステートの表示は、ソフトウェア開発プロセスの理解および管理に役立つとともに、現在と以前のコンフィグレーション構造の全体像が明確になります。

SCM は、コンフィグレーションステートの定義、操作、視覚化をサポートしています。ステートおよびステートの移行は、既存の開発プロセスに最適かつ柔軟に統合できるように、ユーザー定義できます。

コマンド構文：

```
escm newstate [-u] -s <サーバ名> -d <リポジトリ名> -p <プロジェクト名>
      -z <ステート名> [-po <ポート>]
```

`-u`
新しいステートを一意として定義します。

「一意」としてステートを設定すると、SCM は常に、このステートになるコンフィグレーションをリリース内で 1 つだけになるように管理します。新たに別のコンフィグレーションにこのステートが割り当てられた場合、SCM は元のコンフィグレーションのステートを自動的に「archived」に変更します。

例：現在、「Build 3」というコンフィグレーションが、一意と設定されているステートである「quality assurance」にあります。開発チームは、いくつかの問題を修正した後、新しいスナップショット（「Build 4」）を作成して品質保証チームに渡します。スナップショットのターゲットステートを「quality assurance」に設定すると、「Build 3」のステートは自動的に「archived」に設定されます。

`-s <サーバ名>`
リポジトリが存在するサーバの名前。

`-d <リポジトリ名>`
プロジェクトが含まれているリポジトリの名前。

`-p <プロジェクト名>`
プロジェクトの名前。

`-z <ステート名>`
新しく作成するステートの名前。

`-po <ポート>`
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、`9500` です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

例

新しいコンフィグレーションステートとして、「pre-delivery」をプロジェクトに追加します。

```
C:¥>escm newstate -s estescmtstxp -d Sample -p "Project A"
-z pre-delivery
```

2.3.12 exportstates – ステート定義のエクスポート [SCM]

SCM プロジェクトのすべての定義済みコンフィグレーションステートを、XML ファイルにエクスポートします。このファイルは他の SCM プロジェクトにインポートしたり、新規プロジェクトのテンプレートとして利用したりすることができます。

コマンド構文：

```
escm {exportstates | exs} -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> [-po <ポート> -t <エクスポート先ファイル名>]
```

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
プロジェクトが含まれているリポジトリの名前。

-p <プロジェクト名>
プロジェクトの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-t <エクスポート先ファイル名>
作成する XML ファイルのパス名。このオプションを省略すると、「<プロジェクト名>_States.xml」という名前のファイルにエクスポートされ、現在の作業フォルダーに保存されます（<プロジェクト名>は、「-p」オプションで指定したプロジェクト名です）。指定したファイルがすでに存在する場合、上書きされます。

例

「Project A」のステート定義をエクスポートします。

```
C:¥>escm exportstates -s estescmtstxp -d Sample -p "Project A"
-t C:¥Tmp¥states.xml
```

以下の内容の XML ファイルが生成されます。

```
<?xml version="1.0" encoding="UTF-8"?>
<states datastore="Sample" project="Project A" server="estescmtstxp">
    <cfgstate name="in development" type="I"/>
    <cfgstate name="archived" type="A"/>
    <cfgstate name="released" type="R"/>
    <cfgstate name="qa" unique="1"/>
</states>
```

2.3.13 importstates – ステート定義のインポート [SCM]

XML ファイルから SCM プロジェクトにコンフィグレーションステートをインポートします。インポートしたステートは、現在の定義済みステートのリストに追加されます。

コマンド構文：

```
escm {importstates | ims} -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -t <インポートする XML ファイル名>
      [-po <ポート>]
```

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
プロジェクトが含まれているリポジトリの名前。

-p <プロジェクト名>
プロジェクトの名前。

-t <インポートする XML ファイル名>
インポートする XML ファイルのパス名。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

例

「C:¥tmp¥states.xml」ファイルの内容をプロジェクトにインポートします。

```
C:¥>escm importstates -s estescmtstxp -d Sample -p "Project A"
-t C:¥Tmp¥states.xml
```

2.3.14 renstate – ステートの名前変更 [SCM]

指定した SCM プロジェクトのコンフィグレーションステートの名前を変更します。

コマンド構文：

```
escm renstate -s <サーバ名> -d <リポジトリ名> -p <プロジェクト名>
      -z <ステート名> [-po <ポート> -prp <プロパティ;...>] <新しいステート名>
```

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
プロジェクトが含まれているリポジトリの名前。

-p <プロジェクト名>
プロジェクトの名前。

-z <ステート名>
変更前のステートの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-prp <プロパティ>
このオプションは今版では利用できません。

<新しいステート名>
変更後のステートの名前。

例

ステート名を「qa」から「quality-assurance」に変更します。

```
C:¥>escm renstate -s estescmtstxp -d Sample -p "Project B" -z qa
quality-assurance
```

2.3.15 delstate – ステートの削除 [SCM]

SCM プロジェクトのコンフィグレーションステートを削除します。

注意：SCM の定義済みシステムステートは、名前変更はできますが、削除または変更はできません。

コマンド構文：

```
escm delstate -s <サーバ名> -d <リポジトリ名> -p <プロジェクト名>
      -z <ステート名> [-po <ポート>]
```

-s <サーバ名>
　リポジトリが存在するサーバの名前。

-d <リポジトリ名>
　プロジェクトが含まれているリポジトリの名前。

-p <プロジェクト名>
　ステートを定義するプロジェクトの名前。

-z <ステート名>
　削除するステートの名前。

-po <ポート>
　リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

例

プロジェクトからユーザー定義ステートの「qa」を削除します。

```
C:¥>escm delstate -s estescmtstxp -d Sample -p "Project A" -z qa
```

2.3.16 ignres – 無視されるリソースパターンの定義 [SCM]

無視されるリソースを定義するパターンを表示、拡張、更新します。add コマンドまたは commit コマンドを使用してリポジトリに新しいリソースを追加する時、無視されるリソースはその対象から外れます。複数のパターンを指定する場合、それぞれのパターンを空白で区切る必要があります。

注意：*.tmp などのワイルドカードパターンは、Windows ではエスケープされた二重引用符（¥" ¥"）で、UNIX/Linux では一重引用符（' '）で囲む必要があります。引用符で囲まない場合、Java VM により実ファイル名として処理されます。

コマンド構文：

```
escm ignres [-l -o] -s <サーバ名> -d <リポジトリ名> -p <プロジェクト名>
      [-po <ポート>] [<パターン 1> ... <パターン n>]
```

-l
　プロジェクトに現在定義されているパターンのリストを表示します。

-o
　プロジェクトの現在の定義を、指定したパターンと置換します。このオプションを指定しない場合、新しい定義が現在のパターンリストに追加されます。

-s <サーバ名>
　リポジトリが存在するサーバの名前。

-d <リポジトリ名>
　プロジェクトが含まれているリポジトリの名前。

-p <プロジェクト名>
　プロジェクトの名前。

-po <ポート>
　リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、

9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

<パターン x>
任意のファイル名のパターン（ワイルドカード*（1 つ以上の文字）および?（1 文字）がサポートされます）。

例

無視するリソースパターンのリストに「*.tmp」というパターンを追加します。ファイル名の拡張子が「tmp」であるすべてのファイルが無視されます。

```
C:¥>escm ignres -s estescmtstxp -d Sample -p "Project A" '*.tmp'
```

プロジェクトに現在定義されているパターンを表示します。

```
C:¥>escm ignres -s estescmtstxp -d Sample -p "Project A" –l
```

2.3.17 setprojectprops – 厳密な変更制御の設定 [SCM、DMS]

SCM プロジェクトまたは DMS ライブラリにおいて厳密な変更制御を有効とするか無効とするかを設定します。

コマンド構文：

```
escm {setprojectprops | spp} -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -z <on/off> [-po <ポート>]
```

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
プロジェクトまたはライブラリが含まれているリポジトリの名前。

-p <プロジェクト名>
更新するプロジェクトまたはライブラリの名前。

-z <on/off>
「on」を指定すると、厳密な変更制御が有効になります。「off」を指定すると、無効になります。有効にした場合、ユーザーがプロジェクトまたはライブラリにリソースをコミットまたは追加するときに、最低 1 つのタスクを指定する必要があります。タスクが割り当てられていないユーザーは、コミットおよび追加機能を使用できません。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

例

「Project B」について厳密な変更制御を有効にします。

```
C:¥>escm setprojectprops -s estescmtstxp -d Sample -p "Project B" -z on
```

2.3.18 list – プロジェクトとリリース、またはライブラリの一覧表示 [SCM、DMS]

SCM プロジェクト、リリース、およびコンフィグレーション、または DMS ライブラリを一覧表示します。

特定のリポジトリ内で有効なすべての SCM プロジェクト、リリース、およびコンフィグレーションまたは DMS ライブラリを一覧表示するには、「-d」オプションと「-s」オプションを指定します。さらに「-p 」オプションを指定すると、特定のプロジェクトまたはライブラリの内容だけが表示されます。「-p」オプションと「-r」オプションの両方を指定すると、特定のプロジェクトの特定のリリースの内容だけが表示されます。「-h」オプションと「–v」オプションを両方とも指定しない場合、非表示設定のプロジェクトを含むすべてのプロジェクトが出力されます。

コマンド構文：

```
escm {list | ls} [-h | –v] [-lop] -s <サーバ名> -d <リポジトリ名>
      [-po <ポート> -p <プロジェクト名> -r <リリース名>
      -ikey <インデックスキー> -ival <インデックス値>]
```

-h
: 非表示設定のプロジェクト（SCM プロダクトなど）だけを一覧表示します。

-v
: 表示設定のプロジェクト（標準プロジェクトまたはライブラリ）だけを一覧表示します。

-lop
: リリースとコンフィグレーションの情報は表示せず、プロジェクト名とプロパティだけを一覧表示します。

-s <サーバ名>
: リポジトリが存在するサーバの名前。

-d <リポジトリ名>
: プロジェクトまたはライブラリが含まれているリポジトリの名前。

-po <ポート>
: リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「カスタマイズガイド」を参照してください。

-p <プロジェクト名>
: リストを生成するプロジェクトまたはライブラリの名前。

-r <リリース名>
: リストを生成するリリースの名前。

-ikey <インデックスキー>, ival <インデックス値>
: このオプションは今版では利用できません。

例

リポジトリのすべてのプロジェクトとプロダクトを表示します。

```
C:¥>escm list -s estescmtstxp -d Sample
```

2.3.19 report – レポートの作成 [SCM]

指定した SCM プロジェクトまたはリポジトリの名前で、レポートを生成します。引数にレポート名を指定しないと、すべての使用可能なレポートが一覧表示されます。

コマンド構文：

```
escm {report | rpt} [-z –z2 –vd –rt –dc -xmlonly] -s <サーバ名>
      -d <リポジトリ名> [-po <ポート> -p <プロジェクト名> -r <リリース名>
      -r2 <リリース名> -c <コンフィグレーション名>
```

```
-c2 <コンフィグレーション名> -t <タスク> -i <アイテムパス>
-f <出力ファイル名> -g <開始日> -h <終了日>
-x <XSL スタイルシート名> -xe <出力文字コード> -b <翻訳ファイル名>
-fa <アクション名> -fu <ユーザー名> -fn <名前> -fp <パス>]
[<レポート名>]
```

-z

以下のレポートに適用されるレポート固有のフラグ

リリースプロパティレポート/コンフィグレーションプロパティレポート/フォルダープロパティレポート/ファイルプロパティレポート

このオプションを指定すると、ファイルやソースコード情報などの追加情報がレポートに含まれます。ファイルスキャンを無効にするには、「-z2」オプションを指定します。

ファイルヒストリレポート

このオプションを指定すると、指定したバージョンの先行バージョンの情報（ファイルヒストリ）だけが、レポートに追加されます。このオプションを指定しないと、すべてのバージョンが含まれます。

リリース変更レポート/コンフィグレーション変更レポート/フォルダー変更レポート/ファイル変更レポート

このオプションを指定すると、作成済みバージョン数に、削除した中間バージョンが含まれます。ただし、レポート生成に時間がかかることがあるので注意してください。

-z2

以下のレポートに適用されるレポート固有のフラグ

リリースプロパティレポート/コンフィグレーションプロパティレポート/フォルダープロパティレポート/ファイルプロパティレポート/リリース変更レポート/コンフィグレーション変更レポート/フォルダー変更レポート/ファイル変更レポート

このオプションを指定すると、ファイルコンテンツのスキャンが無効になります。

-vd

以下のレポートに適用されるレポート固有のフラグ

リリース変更レポート/コンフィグレーション変更レポート/フォルダー変更レポート/ファイル変更レポート

このオプションを指定すると、変更された各バージョンの詳細情報がレポートに含まれます。

-rt

以下のレポートに適用されるレポート固有のフラグ

コンフィグレーション変更レポート(日付別のコンフィグレーション変更レポートは含みません)/コンフィグレーションプロパティレポート

このオプションを指定すると、解決済みのタスクの詳細情報がレポートに含まれます。

-dc

内部レポートキャッシュを無効にします。通常、このオプションを使用するとレポート生成のパフォーマンスが低下しますが、すべてのレポート結果が確実に再計算されます（何らかの理由でキャッシュが壊れた場合などに使用します）。

-xmlonly
スタイルシートや翻訳ファイルを使用しないで、XML 形式のレポートを出力します。

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
プロジェクトが含まれているリポジトリの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-p <プロジェクト名>
プロジェクトの名前（レポートの種類により、必要な場合のみ指定します）。

-r <リリース名>
リリースの名前（レポートの種類により、必要な場合のみ指定します）。

-r2 <リリース名>
2 番目のリリースの名前（変更レポートの場合など）。

-t <タスク>
LCM タスクの ID（タスクの変更レポートに対してのみ必要）。ID の前部にパディングされている「0」は、省略できます。

-c <コンフィグレーション名>
コンフィグレーションの名前（レポートの種類により、必要な場合のみ指定します）。

-c2 <コンフィグレーション名>
2 番目のコンフィグレーションの名前（変更レポートの場合など）。

-i <アイテムパス>
アイテムのパス名（レポートの種類により、必要な場合のみ指定します）。

-f <出力ファイル名>
出力ファイルのパス名。外部スタイルシート(「-x」オプション)または「-xmlonly」オプションを使用して異なる出力形式を指定する場合を除き、出力ファイルの拡張子は.html にする必要があります。<レポート名>を指定する場合、このオプションは必須です。指定したファイルがすでに存在する場合、上書きされます。

-g <開始日>; -h <終了日>
yyyy-MM-dd 形式で記述される開始日と終了日（日付別のログレポートや変更レポートなど、レポートの種類に必要な場合のみ）。

-x <XSL スタイルシート名>
出力生成用のスタイルシートのパス名。省略すると、デフォルトの SCM スタイルシートが使用されます。

-xe <出力文字コード>
出力ファイルの文字コード。このオプションを指定しないと、XML スタイルシート(「-x」オプション)に定義されている文字コードが使用されます。文字コードの例：Shift_JIS、EUC_JP、UTF-8

-b <翻訳ファイル名>
出力生成用の翻訳ファイルのパス名。省略すると、デフォルトの SCM 翻訳ファイルが使用されます。

-fa <アクション名>
プロジェクトログレポートのみ指定可能です：指定したアクションによってログエ

ントリーをフィルタリングします。<アクション名>に、アクション名をセミコロンで区切って列挙します。アクション名は利用可能なオーソリティの名前になります。利用可能なすべてのオーソリティは auth コマンドを使って表示します。

-fu <ユーザー名>
プロジェクトログレポートのみ指定可能です：指定したユーザーによってログエントリーをフィルタリングします。<ユーザー名>に、ユーザー名をセミコロンで区切って列挙します。このオプションを使用できるのは、特権ユーザーだけです。

-fn <名前>
プロジェクトログレポートのみ指定可能です：指定した文字列によってログエントリーをフィルタリングします。<名前>に、文字列をセミコロンで区切って列挙します。ワイルドカード「*」（1 つ以上の文字）および「?」（1 文字）を文字列内に使用できます。

-fp <パス>
プロジェクトログレポートのみ指定可能です：指定したパス文字列によってログエントリーをフィルタリングします。<パス>に、パスをセミコロンで区切って列挙します。ワイルドカード「*」（1 つ以上の文字）および「?」（1 文字）を文字列内に使用できます。パスセパレーター（/または¥）の相違は無視されます。

<レポート名>
生成するレポートの名前。名前を指定しない場合、指定したエンティティのすべての使用可能なレポートのリストが表示されます。指定するレポートの名前は、言語によって異なります。ご使用されている言語で指定する必要があります。レポートの名前を指定する場合、-f オプションを指定する必要があります。

例

リポジトリとプロジェクトについて、すべてのレポートを表示します。

```
C:¥>escm report -s estescmtstxp -d Sample -p "Project B"
```

2.4 ロールとオーソリティ

SCM および DMS アクセス制御システムは、ユーザーのロールに基づいています。これにより、特権ユーザーに対して、各プロジェクトまたはライブラリのロールセットを管理するための管理機能が提供されます。ロールを定義して、ロールにオーソリティを割り当てることで、特定の機能へのアクセスを許可または拒否することができます。また、SCM と DMS には、ロールを定義して管理する機能と、ユーザーにロールを割り当てる機能、またはロールにユーザーを割り当てる機能が提供されています。

SCM または DMS から LCM へのユーザーとロールの割当ての反映

SCM または DMS のユーザーとロールの割当ての変更を LCM に反映すると、対応する LCM のオーソリティを持つ必要なロールをユーザーに割り当てます。

LCM に反映できる SCM または DMS のオーソリティと、対応する LCM のオーソリティを以下の表に記載しています。

※斜体文字はオーソリティの内部名です。

SCM のオーソリティ	DMS のオーソリティ	対応する LCM のオーソリティ
プロジェクトの登録 *ACTION_PRJ_REGISTER*	ライブラリの作成 *ACTION_LIB_CREATE*	アプリケーションの作成
プロジェクトのロールの割当て *ACTION_PRJ_ASSIGNROLES*	ライブラリのロールの割当て *ACTION_LIB_ASSIGNROLES*	アプリケーションの編集
プロジェクトの名前変更 *ACTION_PRJ_RENAME*	ライブラリの名前変更 *ACTION_LIB_RENAME*	アプリケーションの編集
プロジェクトの削除 *ACTION_PRJ_DELETE*	ライブラリの削除 *ACTION_LIB_DELETE*	アプリケーションの削除
フォローアップリリースの作成 *ACTION_REL_NEWFOLLOWUP*	- -	リリースの編集
ブランチリリースの作成 *ACTION_REL_NEWBRANCH*		リリースの編集
リリースの名前変更 *ACTION_REL_RENAME*		リリースの編集
リリースを閉じる *ACTION_REL_CLOSE*		リリースの編集
リリースの削除 *ACTION_REL_DELETE*		リリースの削除
閉じたリリースの再活性化 *ACTION_REL_REACTIVATE*		リリースの編集
ブランチの作成 *ACTION_CFG_NEWBRANCH*		リリースの編集
コンフィグレーションの名前変更	エディションの名前変更 *ACTION_EDT_RENAME*	リリースの編集

SCM のオーソリティ	DMS のオーソリティ	対応する LCM のオーソリティ
ACTION_CFG_RENAME		
コンフィグレーションの削除 *ACTION_CFG_DELETE*	エディションの削除 *ACTION_EDT_DELETE*	リリースの編集
コミットするリソースとしてマーク *ACTION_MARK_FOR_COMMIT*	チェックインするリソースとしてマーク *ACTION_MARK_FOR_COMMIT*	アプリケーションの表示 リリースの表示 タスクの表示 タスクの編集
コミット *ACTION_RES_COMMIT*		アプリケーションの表示 リリースの表示 タスクの表示 要件の表示 タスクの作成 タスクの編集
コミット (上書き) *ACTION_RES_COMMIT_OVR*	チェックイン *ACTION_DOC_COMMIT*	アプリケーションの表示 リリースの表示 タスクの表示 要件の表示 タスクの作成 タスクの編集

2.4.1 auth – オーソリティの表示 [SCM、DMS]

SCM または DMS で使用可能なすべてのオーソリティを表示します。表示されるオーソリティは、オーソリティの論理名のアルファベット順で並び替えられます。

表示されるオーソリティには、管理アクションとしてマークされているものがあります。これらのアクションは、オーソリティの設定とは関係なく、リポジトリの機能によって直接チェックされるので、これらのオーソリティをロールに割り当てる必要はありません。

コマンド構文：

```
escm {auth | a} [-dms -int]
```

`-dms`
SCM オーソリティではなく、DMS オーソリティを表示します。

`-int`
表示名を含むオーソリティを表示します。各オーソリティは、技術名と論理名で一覧表示されます。オーソリティは、論理名のアルファベット順で並び替えられます。

例

既存のすべてのオーソリティを表示します。

```
C:¥>escm auth -int
```

2.4.2 roles – すべての定義済みロールの表示 [SCM、DMS]

SCM プロジェクトまたは DMS ライブラリのすべての定義済みロールを表示します。

コマンド構文：

```
escm {roles | r} -s <サーバ名> -d <リポジトリ名> -p <プロジェクト名>
      [-po <ポート>]
```

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
プロジェクトまたはライブラリが含まれているリポジトリの名前。

-p <プロジェクト名>
プロジェクトまたはライブラリの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

例

「Project B」のすべての定義済みロールを表示します。

```
C:¥>escm roles -s estescmtstxp -d Sample -p "Project B"
```

2.4.3 copyrole – 新規ロールへの既存ロールのコピー [SCM、DMS]

SCM プロジェクトまたは DMS ライブラリに、指定した既存ロールと同じオーソリティを持つロールを作成します。

コマンド構文：

```
escm {copyrole | cr} -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -z <ロール名> -e <新規ロール名> [-po <ポート>]
```

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
プロジェクトまたはライブラリが含まれているリポジトリの名前。

-p <プロジェクト名>
ロールを生成するプロジェクトまたはライブラリの名前。

-z <ロール名>
コピーする既存ロールの名前。

-e <新規ロール名>
作成する新規ロールの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、

9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

例

既存ロール「Developer」と同一の新規ロール「Junior Developer」を作成します。

```
C:¥>escm copyrole -s estescmtstxp -d Sample -p "Project B" -z Developer
-e "Junior Developer"
```

2.4.4 newrole – ロールの作成 [SCM、DMS]

SCM プロジェクトまたは DMS ライブラリ用のロールを作成します。作成直後には、ロールにオーソリティは定義されていません。

コマンド構文：

```
escm {newrole | nr} -s <サーバ名> -d <リポジトリ名> -p <プロジェクト名>
      -z <ロール名> [-po <ポート>]
```

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
プロジェクトまたはライブラリが含まれているリポジトリの名前。

-p <プロジェクト名>
ロールを生成するプロジェクトまたはライブラリの名前。

-z <ロール名>
作成するロールの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

例

プロジェクトに「Senior developer」という新規ロールを追加します。

```
C:¥>escm newrole -s estescmtstxp -d Sample -p "Project B" -z "Senior
Developer"
```

2.4.5 delrole – ロールの削除 [SCM、DMS]

SCM プロジェクトまたは DMS ライブラリから既存のロールを削除します。

コマンド構文：

```
escm {delrole | dr} -s <サーバ名> -d <リポジトリ名> -p <プロジェクト名>
      -z <ロール名> [-po <ポート>]
```

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
プロジェクトまたはライブラリが含まれているリポジトリの名前。

-p <プロジェクト名>
ロールが定義されているプロジェクトまたはライブラリの名前。

-z <ロール名>
削除するロールの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

例

プロジェクトからのロール「Junior Developer」を削除します。

```
C:¥>escm delrole -s estescmtstxp -d Sample -p "Project B" -z "Junior
Developer"
```

2.4.6 exportroles – ロール定義のエクスポート [SCM、DMS]

SCM プロジェクトまたは DMS ライブラリのすべてのロール定義を XML ファイルにエクスポートします。このファイルは他の SCM プロジェクトや DMS ライブラリにインポートしたり、新規プロジェクトやライブラリのテンプレートとして使用したりすることができます。

コマンド構文：

```
escm {exportroles | exr} [-u -dms] -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> [-po <ポート> -t <エクスポート先ファイル名>]
```

`-u`
: すべてのユーザー/ロールの割り当てを同時にエクスポートします。

`-dms`
: SCM オーソリティではなく、DMS オーソリティを持つロールをエクスポートします。

`-s <サーバ名>`
: リポジトリが存在するサーバの名前。

`-d <リポジトリ名>`
: プロジェクトまたはライブラリが含まれているリポジトリの名前。

`-p <プロジェクト名>`
: プロジェクトまたはライブラリの名前。

`-po <ポート>`
: リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、`9500` です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

`-t <エクスポート先ファイル名>`
: 作成する XML ファイルのパス名。このオプションを省略すると、「<プロジェクト名>_Roles.xml」という名前のファイルにエクスポートされ、現在の作業フォルダーに保存されます（<プロジェクト名>は、「-p」オプションで指定したプロジェクト名です）。指定したファイルがすでに存在する場合、上書きされます。

例

「Project B」のロール定義およびすべてのユーザー割り当て情報をエクスポートします。

```
C:¥>escm exportroles -s estescmtstxp -d Sample -p "Project B" -u
-t C:¥tmp¥roles.xml
```

以下の内容の XML ファイルが生成されます（途中省略）。

```
<?xml version="1.0" encoding="UTF-8"?>
<roles datastore="Sample" project="Project B" server="estescmtstxp">
    <role default="1" name="Developer">
        <authority name="ACTION_RES_COMMIT"/>
        <authority name="ACTION_RES_UPDATE"/>
        <authority name="ACTION_RES_COMMIT_OVR"/>
        <authority name="ACTION_RES_UPDATE_OVR"/>
        ...
```

```
    </role>
    ...
    <user name="ESTSCMTSTXP¥Test">
        <assignedrole name="Developer"/>
        ...
    </user>
    ...
</roles>
```

2.4.7 importroles – ロール定義のインポート [SCM、DMS]

XML ファイルまたは指定したプロジェクトやライブラリから、ロール定義またはユーザーとロール間の割り当て情報のどちらか、またはその両方を SCM プロジェクトや DMS ライブラリにインポートします。インポートしたロールおよび割り当て情報は、現在の定義リストに追加されます。

コマンド構文：

```
escm {importroles | imr} [-u -m -dms] -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名>
      {-e <ソースプロジェクト名> | -t <インポートする XML ファイル名>}
      [-po <ポート>]
```

-u
: 指定したソースプロジェクト、ソースライブラリ、または XML ファイルから、ロールに割り当てられているすべてのユーザーをインポートします。

-m
: インポートしたロール定義を、既存の定義と置換するのではなく、既存の定義とマージします。

-dms
: SCM オーソリティではなく、DMS オーソリティを持つロールをインポートします。

-s <サーバ名>
: リポジトリが存在するサーバの名前。

-d <リポジトリ名>
: プロジェクトまたはライブラリが含まれているリポジトリの名前。

-p <プロジェクト名>
: プロジェクトまたはライブラリの名前。

-e <ソースプロジェクト名>
: インポート対象ロールが含まれているソースプロジェクトまたはソースライブラリの名前。

-t <インポートする XML ファイル名>
: インポートする XML ファイルのパス名。

-po <ポート>
: リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

例

「C:¥tmp¥roles.xml」ファイルをプロジェクトにインポートします。

```
C:¥>escm importroles -s estescmtstxp -d Sample -p "Project B"
-t C:¥tmp¥roles.xml
```

2.4.8 rmauthority – ロールからのオーソリティの削除 [SCM、DMS]

SCM プロジェクトまたは DMS ライブラリ内の指定ロールからオーソリティを削除します。

使用可能なすべてのオーソリティ名のリストを表示するには、auth コマンドを使用します。

コマンド構文：

```
escm {rmauthority | rma} [-dms] -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -z <ロール名> -a <オーソリティ名>
      [-po <ポート>]
```

-dms
SCM オーソリティではなく、DMS オーソリティを削除します。

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
プロジェクトまたはライブラリが含まれているリポジトリの名前。

-p <プロジェクト名>
ロールが定義されているプロジェクトまたはライブラリの名前。

-z <ロール名>
変更するロールの名前。

-a <オーソリティ名>
ロールから削除するオーソリティの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

例

「Junior Developer」というロールから「Mark as Merged」というオーソリティを削除します。

```
C:¥>escm rmauthority -s estescmtstxp -d Sample -p "Project B"
-z "Junior Developer" -a "Mark As Merged"
```

2.4.9 setauthority – ロールへのオーソリティの追加 [SCM、DMS]

SCM プロジェクトまたは DMS ライブラリのロールにオーソリティを追加します。

使用可能なすべてのオーソリティ名のリストを表示するには、auth コマンドを使用します。

コマンド構文：

```
escm {setauthority | sa} [-dms] -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -z <ロール名> -a <オーソリティ名> [-po <ポート>
      -prp <プロパティ;...>]
```

-dms
SCM オーソリティではなく、DMS オーソリティを追加します。

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
プロジェクトまたはライブラリが含まれているリポジトリの名前。

-p <プロジェクト名>
ロールが定義されているプロジェクトまたはライブラリの名前。

-z <ロール名>
変更するロールの名前。

-a <オーソリティ名>
ロールに追加するオーソリティの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-prp <プロパティ;...>

このオプションは今版では利用できません。

例

「Developer」というロールに「Create Branch」というオーソリティを追加します。

```
C:¥>escm setauthority -s estescmtstxp -d Sample -p "Project B"
-z Developer -a "Create Branch"
```

2.4.10 roleauth – ロール情報の表示 [SCM、DMS]

SCM プロジェクトまたは DMS ライブラリのロールの定義済みオーソリティをすべて表示します。

コマンド構文：

```
escm {roleauth | ra} [-dms] -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -z <ロール名> [-po <ポート>]
```

-dms
SCM オーソリティではなく、DMS オーソリティを表示します。

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
プロジェクトまたはライブラリが含まれているリポジトリの名前。

-p <プロジェクト名>
ロールが定義されているプロジェクトまたはライブラリの名前。

-z <ロール名>
表示するロールの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

例

「Junior Developer」というロールの定義情報を表示します。

```
C:¥>escm roleauth -s estescmtstxp -d Sample -p "Project B"
-z "Junior Developer"
```

2.4.11 assignrole – ユーザーへのロールの割当て [SCM、DMS]

SCM プロジェクトまたは DMS ライブラリのユーザーにロールを割り当てます。

変更管理が有効で、SCM リポジトリとの関連付けが設定されていれば、SCM プロジェクトまたは DMS ライブラリから LCM アプリケーションに、ロールの割当てを反映させることができます。LCM アプリケーションでは、必要に応じてコマンドを実行したユーザーに 1 つ以上のユーザーロールが割り当てられます。これは、ユーザーに割り当てられている SCM ユーザーロールから、関連するすべてのオーソリティを LCM オーソリティにマップすることで実施します。関連するオーソリティは、SCM アクションを LCM に反映させるために必要となるオーソリティです。

SCM または DMS と LCM 間の関連するオーソリティのマッピングの詳細については、51 頁の「ロールとオーソリティ」を参照してください。

コマンド構文：

```
escm {assignrole | asr} [-nolcm] -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -u <ユーザー名> -z <ロール名> [-po <ポート>]
```

`-nolcm`
LCM への接続を無効にします。この場合、ロールの割当ては反映されません。「変更管理への自動反映」がリポジトリで有効になっている場合は（`chmansett` コマンドを参照）、このオプションで接続を無効にできません。

`-s <サーバ名>`
リポジトリが存在するサーバの名前。

`-d <リポジトリ名>`
プロジェクトまたはライブラリが含まれているリポジトリの名前。

`-p <プロジェクト名>`
ユーザーにロールを割り当てるプロジェクトまたはライブラリの名前。

`-u <ユーザー名>`
ユーザーの名前。

`-z <ロール名>`
ユーザーに割り当てるロールの名前。

`-po <ポート>`
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、`9500` です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

例

プロジェクト「Demo」内のユーザー「estenabler1¥sampleuser2」に、「Project Manager」という追加ロールを定義します。

```
C:¥>escm assignrole -s estenabler1 -d SCM1 -p Demo
   –u estenabler1¥sampleuser2 -z "Project manager"
```

2.4.12 withdrawrole – ユーザーのロール割当て解除 [SCM、DMS]

SCM プロジェクトまたは DMS ライブラリのユーザーから、ロールの割当てを解除します。

変更管理が有効で、SCM リポジトリとの関連付けが設定されていれば、SCM プロジェクトまたは DMS ライブラリから LCM アプリケーションに、ロール割当ての変更を反映させることができます。LCM アプリケーションでは、必要に応じてコマンドを実行したユーザーに 1 つ以上のユーザーロールが割り当てられます。これは、ユーザーに割り当てられている SCM ユーザーロールに含まれるすべての関連するオーソリティを LCM オーソリティにマップすることで実施します。関連するオーソリティは、SCM アクションを LCM に反映させるために必要となるオーソリティです。

SCM または DMS と LCM 間の関連するオーソリティのマッピングの詳細については、51 頁の「ロールとオーソリティ」を参照してください。

コマンド構文：

```
escm {withdrawrole | wdr} [-nolcm] -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -u <ユーザー名> -z <ロール名> [-po <ポート>]
```

`-nolcm`
LCM への接続を無効にします。この場合、ロール割当ての変更は反映されません。「変更管理への自動反映」がリポジトリで有効になっている場合は（`chmansett` コマンドを参照）、このオプションで接続を無効にできません。

`-s <サーバ名>`
リポジトリが存在するサーバの名前。

`-d <リポジトリ名>`
プロジェクトまたはライブラリが含まれているリポジトリの名前。

`-p <プロジェクト名>`
ユーザーのロール割当てを解除するプロジェクトまたはライブラリの名前。

`-u <ユーザー名>`
ユーザーの名前。

`-z <ロール名>`
ユーザーから割当てを解除するロールの名前。

`-po <ポート>`
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、`9500` です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

例

プロジェクト「Demo」内のユーザー「estenabler1¥sampleuser2」から、「Project Manager」というロールを削除します。

```
C:¥>escm withdrawrole -s estenabler1 -d Sample -p Demo
   –u estenabler1¥sampleuser2 -z "Project manager"
```

2.4.13 userroles – ユーザーロールの表示 [SCM、DMS]

SCM プロジェクトまたは DMS ライブラリのユーザーに割り当てられているロールを表示します。（「`-u`」オプションで）ユーザー名を指定しない場合、実行ユーザーのロールのリストが表示されます。

コマンド構文：

```
escm {userroles | ur} -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> [-po <ポート> -u <ユーザー名>]
```

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
プロジェクトまたはライブラリが含まれているリポジトリの名前。

-p <プロジェクト名>
ユーザーのロールを表示するプロジェクトまたはライブラリの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-u <ユーザー名>
ロールを表示するユーザーの名前。指定しないと、実行ユーザーに割り当てられているロールのリストが表示されます。

例

プロジェクト「Demo」内のユーザー「estenabler1¥sampleuser2」に割り当てられているロールを表示します。

```
C:¥>escm userroles -s estenabler1 -d Sample -p Demo
   –u estenabler1¥sampleuser2
```

2.4.14 users – ロール割当てユーザーのリスト [SCM、DMS]

SCM プロジェクトまたは DMS ライブラリ内で、少なくとも 1 つのロールが割り当てられているユーザーのリストを表示します。明示的にロールが割り当てられているユーザーと、プロジェクトやライブラリに定義済みのデフォルトロールが割り当てられているユーザーが表示されます。

コマンド構文：

```
escm {users | u} -s <サーバ名> -d <リポジトリ名> -p <プロジェクト名>
      [-po <ポート>]
```

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
プロジェクトまたはライブラリが含まれているリポジトリの名前。

-p <プロジェクト名>
ユーザーのリストを表示するプロジェクトまたはライブラリの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

例

プロジェクト「Demo」内のロール割り当てユーザーのリストを表示します。

```
C:¥>escm users -s estenabler1 -d Sample -p Demo
```

2.5 リソースの操作

SCM プロジェクト内のリソースを管理するには、以下のコマンドを使用します。一部のコマンドは、DMS ライブラリのフォルダーやドキュメントにも適用できます。

変更管理が有効でリポジトリ用に設定されていれば、SCM および DMS で、コミットと追加の操作によりタスクの仕様をサポートできます。変更管理は、開発作業の進行状況を追跡して監査したり、関連アクティビティをまとめたりしたい場合に、特に役立ちます。

2.5.1 search – リソースの検索 [SCM, DMS]

ファイル内のテキストやコメントを検索します。または検索設定の有効性を確認します。

SCM のプロジェクト、リリース、コンフィグレーション、およびフォルダーやファイル、DMS のライブラリ、フォルダー、ファイルを、指定した文字列や条件で検索します。同じリポジトリ内であれば、複数のプロジェクトやライブラリを検索できます。

検索をするには、検索を可能に設定し、検索サーバとの接続を確立する必要があります。詳細については、searchsett コマンドを参照してください。さらに、検索インデックスを検索サーバに作成しておかなければなりません。詳細については、「管理者ガイド」を参照してください。

「-p」、「-r」、「-c」、「-i」のオプションを使用して、検索の範囲を設定できます。これらのオプションは、複数の値を持つことができます。値の区切りには、セミコロンを使用してください。また、各値をクォーテーションで囲む必要があります。これらのオプションのうち 1 つを使用した場合、それより下位の範囲を指定するオプションは指定できません。具体的には、「-p」オプションを使用してプロジェクトを指定した場合、「-r」、「-c」、「-i」は使用できません。

検索結果は、検索語を含むか指定した条件に一致するファイルのリストとして出力されます。検索開始時に選択したアイテムによって、検索結果が異なる可能性があります。Lost & Found フォルダーは検索されません。SCM では、中間バージョンは、1 つのプロジェクトを対象とした場合のみ検出されます。

コマンド構文：

```
escm {search | sr} {-v | –cn –cm –x} -s <サーバ名>
      -d <リポジトリ名> [-po <ポート> -p <プロジェクト> -r <リリース>
      -c <コンフィグレーション> -i <アイテム> -st <検索文字列>
      -fp <ファイルパス> -cu <コミットしたユーザー> -fr <開始日>
      -to <終了日> -so <並び替え> -mx <最大検索数>]
```

-v
: リポジトリに対して検索設定を確認し、表示します。確認とは、検索サーバへの接続をテストすることです。リポジトリサーバとリポジトリは、それぞれ「-s」、「-d」オプションで指定する必要があります。「-v」オプションと検索条件を指定するオプションは、同時に実行できないため、互いに排他的です。

-cn
: ファイルやドキュメントの内容を検索します。このオプションに追加して、「-cm」オプションを指定できます。「-cn」、「-cm」オプションの両方が指定されない場合、ファイルやドキュメントの内容のみ検索します。

-cm
: リソースのコミットコメントやドキュメントのチェックインコメントを検索します。このオプションに追加して、「-cn」オプションを指定できます。「-cm」、「-cn」オプションの両方が指定されない場合、ファイルやドキュメントの内容のみ検索します。

-x
コミットやチェックインした日時、ユーザー、コメントを含めて、検索結果を表示します。デフォルトでは、検索結果には、見つかったファイル名、プロジェクト名やライブラリ名、および SCM ではコンフィグレーション名が表示されます。

-s <サーバ名>
リポジトリが存在するサーバ名です。

-d <リポジトリ名>
検索するファイルが存在するリポジトリ名です。

-po <ポート>
リポジトリサーバにアクセスするポート番号です。初期状態でのデフォルトでは 9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-p <プロジェクト名>
検索するプロジェクトやライブラリの名前です。

-r <リリース名>
検索するリリース名です。

-c <コンフィグレーション名>
検索するコンフィグレーション名です。

-i <アイテム>
検索するフォルダーやファイルのパスと名前です。

-st <検索文字列>
検索する文字列です。複数の検索語を指定する場合、引用符で囲む必要があります。例えば、”abc 123”を指定した場合、abc と 123 を含むファイルを検索します。検索の規則と構文については以下を参照してください。

-fp <ファイルパス>
検索結果でフィルタリングするファイルパスを指定します。ここで指定したパターンに一致するファイルのみが、検索結果として表示されます。パス区切り文字には、スラッシュ（/）を使用してください。ワイルドカード文字として、任意の文字列と置き換えられる*と、任意の 1 文字と置き換えられる?が使用できます。
例えば、/myfolder/*.java と指定すると、myfolder フォルダー内の.java 拡張子を含むファイルのみが検索されます。

-cu <コミットしたユーザー>
特定のユーザーがコミットまたはチェックインしたファイルやフォルダーのみを検索します。完全なユーザー名を指定する必要があります。

-fr <開始日>
指定した 日付以降にコミットまたはチェックインされたファイルやフォルダーのみを検索します。日付は、yyyy-MM-dd の形式で指定します。

-to <終了日>
指定した日付以前にコミットまたはチェックインされたファイルやフォルダーのみを検索します。日付は、yyyy-MM-dd の形式で指定します。

-so <並び替え>
検索結果の並び替えを指定します。以下の並び替え条件がサポートされています。
path（デフォルト）、name、user、date
デフォルトは昇順で並び替えられます。降順で並び替えるためには、並び替えの型に desc を追加してください。具体的には次のように指定します。
-so datedesc （日付について降順で表示します。）
検索結果は Unicode で並び替えられるので、ファイル名に日本語が含まれていても並び替えることができます。

`-mx <最大検索数>`
検索結果の最大数です。デフォルトは `20` に設定されています。`-mx all` と指定することで、すべての結果が表示されます。

例

リポジトリ"test"に含まれるプロジェクト"test1"とプロジェクト"test2"内の、文字列"Fujitsu Limited"を検索します。検索結果は、コミットした日時について降順で表示します。

```
C:¥>escm search –x –s testserver –d test –p "test1;test2"
–st "¥"Fujitsu Limited¥"" –so datedesc
```

検索規則

検索は以下の規則に従って実行されます。

ファイルの種類

以下の種類のファイルは検索されます。

- SCM プロジェクトまたは DMS ライブラリのソースファイルとして指定されたすべての種類
- Microsoft Office Word ファイル：`*.doc`, `*.dot`
- Microsoft Office Excel ファイル：`*.xls`, `*.xlb`, `*.xlt`
- Microsoft Office PowerPoint ファイル：`*.ppt`, `*.pps`
- PDF ファイル：`*.pdf`
- HTML ファイル： `*.html`, `*.htm`, `*.shtml`
- XML ファイル：`*.xml`, `*.xsl`
- RTF ファイル：`*.rtf`

拡張子は、大文字と小文字の区別をしません。

検索構文と操作

「`–st`」オプションでは、以下の構文と操作が使用できます。

- 検索では大文字と小文字を区別しません。例えば、`abc` は、`Abc`、`ABC`、`aBc` を指定しても検索されます。
- 特別な文字はインデックスされないので、検索語に指定できません。
- ワイルドカード：任意の文字列に置き換えできる`*`と、任意の 1 文字に置き換えられる`?`を使用できます。ワイルドカード文字は、検索語の最初の文字には指定できません。

例えば、以下の検索語を指定することで、abcdefg が検索できます。
ab*
 a*g
abc*
abcdef?
a??defg

ワイルドカード文字を使用する場合、以下のことに注意してください。

検索インデックスに存在し、検索語と合った語が非常に多くなる可能性があります。検索語は 1024 までに制限されています。それ以上になった場合、適切なエラーメッセージを表示します。

例：

検索語として ca* を指定したとします。インデックスが、car、cap、cat を含んでいる場合、検索語は car、cap、cat の 3 つになります。

- AND 演算子： 検索語を複数指定することで、検索結果を絞り込みます。例えば、abc AND 123 と指定することで、abc と 123 を含むファイルを検索できます。. AND は大文字で書く必要があります。明示的に AND をつけなくても、デフォルトで設定されています。
- OR 演算子： 検索語のどれかを含むファイルを検索します。例えば、abc OR 123 と指定することで、abc または 123 を含むファイルが検索できます。OR は大文字で書く必要があります。
- NOT 演算子：指定した語を検索から除外します。例えば、abc NOT 123 と指定すると、abc が含まれるファイルの中で、123 が含まれないすべてのファイルを検索します。NOT は大文字で書く必要があります。
- 入れ子：AND と OR 演算子を括弧で結合することで、一度に複数のタスクを実行できます。例えば、abc AND (123 OR xyz)と指定することで、abc を含み、さらに 123、xyz のどちらかを含むファイルを検索できます。
- 制御：マイナス記号（-）を語の先頭につけることで、その語を検索結果に含まないようにすることができます。例えば、Fujitsu -Limited と指定すると、語 Fujitsu を含み、 Limited を含まないファイルのみを検索できます。
- フレーズ：完全な句を検索する場合、検索語をバックスラッシュでエスケープした追加のダブルクォーテーションで囲んでください。例えば、"¥"Fujitsu Limited¥""と指定すると、指定した語順でこの句を含むファイルを検索できます。句が停止語を含む場合、その停止語は検索からは無視されます。例えば、検索語を "¥"Fujitsu Limited¥""と指定すると、Fujitsu and Limited という句が含まれるファイルも検索されます。なぜならば、and は停止語であり、検索インデックスに含まれないためです。

検索語の区切り

検索語は以下の文字と規則で区切られます。

- アルファベット以外の文字：アルファベットと文字以外のすべての文字です。例えば、/ * - + ( )などです。
  これらの文字は検索語内には使用できません。
- 改行
- 空白
- 文字と数字の区切り
- 異なったエンコードの文字：例えば、日本語から英語へ切り替わったときや、日本語内でひらがなからカタカナ、漢字が切り替わったときです。

- 上記の規則が適用されない場合、全角文字と半角文字の両方を含む検索語は 1 つの検索語とみなされます。

区切られた語は、AND で結ばれたものとして扱われます。

例:

検索テキスト	検索語
ABC DEF GHIJK	ABC
	DEF
	GHIJK
abc123def	abc
	123
	def
日本語 ABC	日本語
	ABC

停止語

停止語は、例えば、a や the などの一般的な語です。これらは検索インデックスに含まれないため、検索できません。以下のリストが、検索に使用されるすべての停止語です。

a	the
and	their
are	then
as	there
at	these
be	they
but	this
by	to
for	was
if	will
in	with
into	いう
is	する
it	人物
no	さま
not	すること
of	ため
on	もの
or	おいて
s	なる
such	できる
t	おく
that	

ある

2.5.2 checkout – ローカルワークスペースへのリソースのコピー [SCM]

SCM プロジェクトからローカルワークスペースに、フォルダーやファイルをチェックアウトします。

プロジェクトリソースを変更する場合、最初にローカルワークスペースにリソースをチェックアウトする必要があります。これにより、必要に応じて、ローカル環境に新規プロジェクトが作成され、指定したすべてのリソースバージョンがワークスペースにコピーされます。SCM は、分散型開発プロセスのリソースを制御します。

「-c」オプションでコンフィグレーションを指定せず、指定したリリースに含まれている作業コンフィグレーションが 1 つだけの場合は、そのコンフィグレーションのリソースがチェックアウトされます。指定したリリース内に複数の作業コンフィグレーションがある場合は、「-c」オプションを使用してチェックアウトするコンフィグレーションを指定する必要があります。

「-i」オプションおよびプロジェクトルートへのパス指定を使用して関連アイテム（ファイルまたはフォルダー）を指定することにより、コンフィグレーションの一部だけをチェックアウトできます。部分チェックアウトに「-q」オプションを併用すると、指定したアイテムのすべての親フォルダーが作成されます（デフォルトでは親フォルダーは作成されません）。「-u」オプションを指定すると、リソースは読み取り専用モードでチェックアウトされます。「-t」オプションを使用すると、チェックアウトしたデータを保存するフォルダーを指定できます（親フォルダーが存在している必要があります）。このオプションを指定しないと、リソースは、現在の作業フォルダーのプロジェクトに対応する名前のフォルダーにチェックアウトされます。

チェックアウトデータを削除するには、単純に、ローカルワークスペース内の対応するプロジェクトのフォルダーを削除します。

コマンド構文：

```
escm {checkout | co} [-u –q –mon –cl –f -ut] -s <サーバ名>
      -d <リポジトリ名> -p <プロジェクト名> -r <リリース名> [-po <ポート>
      -c <コンフィグレーション名> -i <アイテムパス>
      -t <格納先フォルダー名>] -fds <開始日> -fde <終了日>]
```

-u
: すべてのチェックアウトリソースを読み取り専用モードに設定します。

-q
: 親フォルダー構造を作成します（部分チェックアウトの場合のみ）。

-mon
: ローカルの変更を制御するリソース監視を作成します。プロジェクト設定でリソース監視が強制されている場合は、「–mon」オプションを指定しなくてもリソース監視が作成されます。

-cl
: 必要に応じて、テキストファイルの改行コードをローカルシステムの値に変換します。このオプションを指定すると、チェックアウト操作により、ローカルで作成されたテキストファイルの改行コードが、必ず、ローカルオペレーティングシステムの改行コードになります。例えば、テキストファイルの改行コードとして、Windows では CR と LF を、UNIX では LF を使用します。

-f
: 指定したフォルダー（「-i」オプション）を、下位構造なしでチェックアウトします。

フォルダーは、ローカルプロジェクトのワークスペース内に部分チェックアウトフォルダーとして作成されます。このオプションは、フォルダーに新しいリソースだけを追加する場合に役立ちます。

`-ut`
現在の操作の日時を、チェックアウトするすべてのファイルリソースに適用します。このオプションを指定しないと、チェックアウトするファイルリソースに、リポジトリに保存されている最終更新日時が適用されます。

`-s <サーバ名>`
リポジトリが存在するサーバの名前。

`-d <リポジトリ名>`
リソースが含まれているリポジトリの名前。

`-p <プロジェクト名>`
リソースが含まれているプロジェクトの名前。

`-r <リリース名>`
リソースが含まれているリリースの名前。

`-po <ポート>`
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、`9500` です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

`-c <コンフィグレーション名>`
チェックアウトするコンフィグレーションの名前。

`-i <アイテムパス>`
チェックアウトするアイテムのパス名。

`-t <格納先フォルダー名>`
チェックアウトデータを格納するフォルダーのパス名。指定しない場合、現在の作業フォルダーが使用されます。

`-fds <開始日>`
特定の期間にコミットされたすべてのリソースをチェックアウトするための開始日。`yyyy-MM-dd` の形式で指定します。終了日は「`-fde`」オプションで指定します。

`-fde <終了日>`
特定の期間にコミットされたすべてのリソースをチェックアウトするための終了日。`yyyy-MM-dd` の形式で指定します。開始日は「`-fds`」オプションで指定します。

例

「MAIN」コンフィグレーションの下位構造をチェックアウトして、親構造を再構築します（親を含む部分チェックアウト）。格納先として、現在の作業フォルダーを使用します。

```
C:¥>escm checkout –q -s estenabler1 -d Sample -p Demo -r 5.00 -c MAIN
-i "/Development/Online Help" –u
```

「MAIN」コンフィグレーションから、下位構造なしでフォルダーをチェックアウトします。格納先として、現在の作業フォルダーを使用します。

```
C:¥>escm checkout -s estenabler1 -d Sample -p Demo -r 5.00 -c MAIN
-i "/Development/Online Help" –f
```

2.5.3 coloc – フォルダーの部分チェックアウトの完了 [SCM]

以前に部分的にチェックアウトしたフォルダーを、完全にチェックアウトします。

コマンド構文：

```
escm coloc [-cl -ut] <フォルダー名> [<フォルダー名> ...]
```

-cl
　必要に応じて、テキストファイルの改行コードをローカルシステムの値に変換します。このオプションを指定すると、チェックアウト操作により、ローカルで作成されたテキストファイルの改行コードが、必ず、ローカルオペレーティングシステムの改行コードになります。例えば、Windows では改行コードとして CR と LF を、UNIX では改行コードとして LF を使用します。

-ut
　現在の操作の日時を、チェックアウトするすべてのフォルダーに適用します。このオプションを指定しないと、チェックアウトするフォルダーに、リポジトリに保存されている最終更新日時が適用されます。

<フォルダー名>
　部分チェックアウトしたフォルダー構造に対してチェックアウトする 1 つ以上のフォルダーのパス名。

例

部分チェックアウト済みのサブフォルダー「Module A」を完全にチェックアウトします。

```
C:¥test¥projects¥Project B>escm coloc "source/Module A"
```

2.5.4 export – リソースのエクスポート [SCM]

リポジトリからローカルファイルシステムに、任意のリソースまたはコンフィグレーション(作業コンフィグレーションまたは保護されたコンフィグレーション)全体をコピーします。この方法で取得したリソースは、エクスポート後は SCM の管理対象ではなくなります。

コマンド構文：

```
escm {export | exp} [-o –m -cl] -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -r <リリース名> -c <コンフィグレーション名>
      [-po <ポート> -i <アイテムパス> -t <格納先フォルダー名>]
```

-o
　ローカルファイルシステムにある既存リソースを上書きします。既存リソースが存在する場合、「-o」オプションを指定しないと、エラーが発生します。

-m
　エクスポート用にマークされているコンフィグレーションやフォルダーのリソースだけをエクスポートします（「mark コマンド」を参照）。単一アイテムをエクスポートする場合は、このオプションは適用されません。

-cl
　必要に応じて、テキストファイルの改行コードをローカルシステムの値に変換します。このオプションを指定すると、エクスポート操作により、ローカルで作成されたテキストファイルの改行コードが、必ず、ローカルオペレーティングシステムの改行コードになります。例えば、Windows では改行コードとして CR と LF を、UNIX では改行コードとして LF を使用します。

-s <サーバ名>
　リポジトリが存在するサーバの名前。

-d <リポジトリ名>
　エクスポートするアイテムが含まれているリポジトリの名前。

-p <プロジェクト名>
エクスポートするアイテムが含まれているプロジェクトの名前。

-r <リリース名>
エクスポートするアイテムが含まれているリリースの名前。

-c <コンフィグレーション名>
エクスポートするアイテムが含まれているコンフィグレーションの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-i <アイテムパス>
エクスポートするアイテムのパス名。このオプションを使用してアイテムを指定しない場合、指定したコンフィグレーションのすべてのリソースがエクスポートされます。

-t <格納先フォルダー名>
リソースのコピーを保存するフォルダーの名前。

例

「source」の下位構造を一時フォルダーにエクスポートします。

```
C:¥>escm export -s estescmtstxp -d Sample -p "Project B" -r 1.0 -c MAIN
-i ¥source -t C:¥Tmp¥PrjSources
```

2.5.5 monitor – ローカルリソース変更の監視 [SCM]

注意: このコマンドは、＜ファイル名/フォルダー名＞の引数を指定しない場合、現在の作業フォルダーに適用されます。

ローカルリソース変更の監視を制御します。

何もオプションを指定しないと、指定したローカルプロジェクトの監視が有効になります（有効でない場合）。

コマンド構文：

```
escm monitor [-sync -remove] [-active <on/off> -m <コメント>]
      [<ファイル名/フォルダー名>]
```

-sync
すべてのローカルリソースの編集状態を再計算し、結果をリポジトリに保存します。

-remove
ローカルプロジェクトの監視を無効にします。他のオプションはすべて無視されます。

-active <on/off>
監視を有効「on」または無効「off」としてマークします。

-m <コメント>
ローカルリソース監視の簡単な説明。

<ファイル名/フォルダー名>
ローカルプロジェクトの一部であるファイルまたはフォルダーの名前。ファイル名とフォルダー名は、絶対パス名または相対パス名を使用して指定します。省略すると、現在の作業フォルダーが使用されます。

例

指定したパスのローカルプロジェクトの監視を有効にします。

```
C:¥>escm monitor "C:¥projects¥workspace¥Project A"
```

指定したパスに存在するすべてのリソースの編集状態を再計算します。

```
C:¥>escm monitor –sync "C:¥projects¥workspace¥Project A"
```

2.5.6 add – バージョン管理へのリソースの追加 [SCM]

注意: このコマンドは、＜ファイル名/フォルダー名＞の引数を指定しない場合、現在の作業フォルダーに適用されます。

リポジトリの作業コンフィグレーション内のローカルリソース（フォルダーおよびファイル）を登録し、バージョン管理の対象とします。同じ名前のリソースがすでに存在する場合、この操作は行えません。

リソースを更新するには、このコマンドを使用して事前にリソースを登録する必要があります。登録されていないリソースは、すべての SCM アクションから無視されます。これには、SCM の対象外とする一時ファイルまたはプライベートファイルの構造をローカルワークスペース内に保存できるという利点があります。

登録済みリソースは、「バージョン管理の対象」となります。

リソースを登録すると、プロジェクトチームのすべてのメンバーが、そのリソースを使用してローカルワークスペースを更新できます。

また、commit コマンドの「–add」オプションを使用して、新しいリソースを登録することもできます。

SCM リポジトリで変更管理が有効に設定されている場合、タスクを指定できます。

コミットコメント

リソースの登録にはリソースのコミットも含まれるので、リソースに関する短いコメントを入力する必要があります。この説明は、リポジトリに格納されます。

リソース名の制限

各種のクライアントオペレーティングシステムを適正にサポートするために、リソースの名前には以下の文字と同様、すべての制御文字を使用できません。
* : ¥ / ? < > | "
リソース名にこれらの文字が含まれていると、リソースの追加に失敗します。

コマンド構文：

```
escm add [-tr -ut] -m <コメント> [-c <コンフィグレーション名>
      -r <リリース名> -t <タスク[;タスク;...]>]
      [<ファイル名/フォルダー名> ...]
```

-tr
: 追加したすべてのリソースの情報をコンソールに出力します。リソースは、完全ローカルパス名で表示されます。

-ut
: 現在の操作の日時を、最終更新日時として、追加するすべてのファイルリソースに適用します。

-m <コメント>
: 新しいリソースバージョンと一緒にリポジトリに保存する短い説明（コミットコメント）。

-c <コンフィグレーション名> / -r <リリース名>
リソースを登録する作業コンフィグレーションを指定します。デフォルトでは、ローカルプロジェクトに割り当てられている作業コンフィグレーションが使用されます。

-t <タスク[;タスク;...]>
新しいリソースを登録する 1 つ以上のタスクの ID を、セミコロンで区切って指定します。ID の前部にパディングされている「0」は、省略できます。

<ファイル名/フォルダー名>
バージョン管理の対象とする 1 つ以上のローカルファイルまたはフォルダーの名前。ファイル名とフォルダー名は混在させることができ、絶対パス名または相対パス名を使用して指定します。省略すると、現在の作業フォルダーが使用されます。

例

「/Documents」のもとで作成された新しい下位構造「QA」を追加します。

```
C:¥>escm add -m "Added QA docums" Documents/QA
```

2.5.7 commit – リポジトリへのローカル変更の格納 [SCM]

注意: このコマンドは、＜ファイル名/フォルダー名＞の引数を指定しない場合、現在の作業フォルダーに適用されます。

ローカルで作成または変更したリソース（フォルダーおよびファイル）を、リポジトリの作業コンフィグレーション内に格納します。ローカルで変更されたファイルは、いずれも新しいバージョンが作成され、作業コンフィグレーション内の前のバージョンと置換されます。

注意: リソースを削除した場合も、そのリソースをリポジトリから削除するには、コミットする必要があります。Windows システムにおいてリソース名の大文字と小文字を変更した場合、新しい（変名された）リソースをコミットする前に、削除された（以前の）リソースをコミットする必要があります。その操作を行わない場合、コミット処理は失敗します。

コミット処理を完了する前に、SCM は、ローカルワークスペース、作業コンフィグレーションの現在のリソースバージョン、およびリソースのベースバージョンをそれぞれ比較します。比較結果に基づいて、コミットアクションでは、ローカルで作成または変更されたリソースだけが確実にリポジトリに転送され、保存されます。デフォルトでは、リポジトリの他のチームメンバーによる変更は上書きされません。この場合、コミット処理に失敗するので、リソースをマージする必要があります。

リソースがコミットされた後、他のチームメンバーは、新しいリソースバージョンを使用してローカルワークスペースを更新できます。

SCM リポジトリで変更管理が有効に設定されている場合、操作時にタスクを指定できます。

コミットコメント

リポジトリに 1 つまたは複数のリソースをコミットするごとに、コミットアクションの理由を示す短いコメントを入力する必要があります。この説明は、対応するファイルバージョンと一緒にリポジトリ内に保存されます。

リソース名の制限

各種のクライアントオペレーティングシステムを適正にサポートするために、リソースの名前には以下の文字と同様、すべての制御文字を使用できません。
* : ¥ / ? < > | "
リソース名にこれらの文字が含まれていると、コミットに失敗します。

コマンド構文：

```
escm {commit | ci} [-o –ul –ct –tr –ut -add] -m <コメント>
      [-c <コンフィグレーション名> -r <リリース名>
      -t <タスク[;タスク;...]>] [<ファイル名/フォルダー名>...]
```

-o
リポジトリの他のユーザーによる変更を、マージしないでローカルリソースに上書きする場合には、「上書きしてコミット」を使用します。

-ul
コミット実行ユーザーによりコミット後に明示的にロックされている、指定したリソースのロックを解除します。暗黙的にロックされていた下位のリソースもロックが解除されます。

-ct
ファイルのタイムスタンプの変更をリソースの変更として扱います。これにより、ファイルの内容が変更されていなくても、更新日時が変更されていればそのファイルは変更されているとみなされ、リポジトリに新しいバージョンとしてコミットできるようになります。このオプションは、ファイルの更新日時を取り扱うアプリケーションの場合に役立ちます。

-tr
コミットされたすべてのリソースの情報をコンソールに出力します。リソースは、完全ローカルパス名で表示されます。各リソースに、コミットによって実行された処理を示すプレフィックスが付けられます。
a – 追加
c – コミット
d – 削除

プレフィックスの後に感嘆符（!）が付いている場合、そのリソースが「上書きしてコミット」によって処理されたことを意味します。

-ut
現在の操作の日時を、最終更新日時として、コミットするすべてのファイルリソースに適用します。

-add
直接指定した、または指定したフォルダーの 1 つに含まれているすべての新しいリソースを、コミットして登録します。これは、add コマンドを使用して新しいリソースを登録する操作と同じです。このオプションを指定しない場合、新しいリソースはコミット操作から除外されます。

-m <コメント>
新しいリソースバージョンと一緒にリポジトリに保存する短い説明（コミットコメント）。

-c <コンフィグレーション名> / -r <リリース名>
変更をコミットする作業コンフィグレーションを指定します。デフォルトでは、ローカルリソースに割り当てられている作業コンフィグレーションが使用されます。

-t <タスク[;タスク;...]>
変更をコミットする 1 つ以上のタスクの ID を、セミコロンで区切って指定します。ID の前部にパディングされている「0」は、省略できます。

<ファイル名/フォルダー名>
リポジトリにコミットする 1 つ以上のローカルファイルまたはフォルダーの名前。ファイル名とフォルダー名は混在させることができ、絶対パス名または相対パス名を使用して指定します。指定したファイルやフォルダーがローカルワークスペース

から削除されている場合、リポジトリのコンフィグレーションからも削除されます。省略すると、現在の作業フォルダーが使用されます。

例

現在の作業フォルダー内のすべての変更をリポジトリにコミットします。

```
C:¥test¥projects¥Project B>escm commit -m "Corrected some spelling
errors."
```

2.5.8 Itemcreq – タスクの割当て、または割当て解除 [SCM、DMS]

SCM プロジェクトまたは DMS ライブラリのファイルバージョンへのタスクの割当てを処理します。

コマンド構文：

```
escm {itemcreq | uic} [-del] -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -r <リリース名> -i <ファイル>
      -v <バージョン> -t <タスク> [-po <ポート>
      -c <コンフィグレーション名>]
```

`-del`
タスクからファイルバージョンへの割当てを解除します。このオプションを指定しない場合、タスクにファイルバージョンが割り当てられます。厳密な変更制御が定義されているプロジェクトまたはライブラリの場合、すべてのファイルバージョンについて、最低 1 つのタスクを割り当てる必要があることに注意してください。

`-s <サーバ名>`
リポジトリが存在するサーバの名前。

`-d <リポジトリ名>`
プロジェクトまたはライブラリが含まれているリポジトリの名前。

`-p <プロジェクト名>`
タスクに割り当てるファイルバージョン、またはタスクからの割当てを解除するファイルバージョンが含まれているプロジェクトまたはライブラリの名前。

`-r <リリース名>`
タスクに割り当てるファイルバージョン、またはタスクからの割当てを解除するファイルバージョンが含まれているリリースの名前。

`-i <ファイル>`
タスクに割り当てるファイル、またはタスクからの割当てを解除するファイルのパス名。

`-v <バージョン>`
タスクに割り当てるファイル、またはタスクからの割当てを解除するファイルのバージョンの数値。

`-t <タスク>`
ファイルバージョンを割り当てる、または割当てを解除するタスクの ID。ID の前部にパディングされている「0」は、省略できます。

`-po <ポート>`
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

`-c <コンフィグレーション名>`
ファイルバージョンが含まれているコンフィグレーションの名前（割当てを解除す

る場合は指定できません）。Lost & Found フォルダーだけに表示されているファイルバージョンの場合は、コンフィグレーション名として「_lostandfound_」を指定します。この場合、Lost & Found フォルダーのファイルパスを指定する必要があります。

例

リリース「2.0」のタスク「0004711」に、「Sample A」プロジェクトのリリース「2.0」の「MAIN」コンフィグレーションに現在表示されているアイテム「/sub 1/TestA.txt」のバージョン 2 を割り当てます。

```
C:¥>escm itemcreq -s soop1 -d Sample -p "Sample A" -r 2.0 -c MAIN
-i "/sub 1/TestA.txt" -v 2 -t 0004711
```

リリース「2.0」のタスク「0004711」から、「/sub 1/TestA.txt」ファイルのバージョン 2 の割当てを解除します。

```
C:¥>escm itemcreq -s soop1 -d Sample -p "Sample A" -r 2.0
-i "/sub 1/TestA.txt" -v 2 -t 0004711 -del
```

2.5.9 update – ローカルワークスペースの更新 [SCM]

注意: このコマンドは、＜ファイル名/フォルダー名＞の引数を指定しない場合、現在の作業フォルダーに適用されます。

リポジトリの作業コンフィグレーションの最新内容を表示して、ローカルリソースを、リポジトリに保存されているバージョンに更新します。

ローカルリソースを更新する前に、SCM は、ローカルワークスペース、作業コンフィグレーションの現在のリソースバージョン、およびそれらのベースバージョンをそれぞれ比較します。

比較結果に基づいて、更新アクションにより、リポジトリ内の変更されたリソースだけがローカルワークスペースに転送されます。デフォルトでは、ローカルワークスペースの変更は上書きされません。この場合、更新に失敗するので、事前にリソースをマージしておく必要があります。

個々のファイルを更新するか、フォルダーまたはプロジェクト全体を選択して、変更されているすべてのリソースを、リポジトリに保存されているリソースによって更新できます。

コマンド構文：

```
escm {update | upd} [-o –ct –ut -cl –tr] [-c <コンフィグレーション名>
      -r <リリース名>] [<ファイル名/フォルダー名> ...]
```

`-o`
ローカルの変更をマージしないで、リポジトリ内の変更によってローカルの変更を上書きするには、「上書きして更新」を使用します。

`-ct`
ファイルのタイムスタンプの変更をリソースの変更として扱います。これにより、ファイルの内容が変更されていなくても、更新日時が変更されていればそのファイルは変更されているとみなされ、リポジトリに新しいバージョンとしてコミットできるようになります。このオプションはオプションは、ファイルの更新日時を取り扱うアプリケーションの場合に役立ちます。

`-ut`
現在の操作の日時を、更新するすべてのファイルリソースに適用します。このオプションを指定しない場合、リポジトリに保存されている最終更新日時が、更新するリソースに適用されます。

-cl
必要に応じて、テキストファイルの改行コードをローカルシステムの値に変換します。このオプションを指定すると、更新操作により、ローカルで作成されたテキストファイルの改行コードが、必ず、ローカルオペレーティングシステムの改行コードになります。例えば、Windows では改行コードとして CR と LF を、UNIX では改行コードとして LF を使用します。

-tr
更新したすべてのリソースの情報をコンソールに出力します。リソースは、完全ローカルパス名で表示されます。各リソースに、更新によって実行された処理を示すプレフィックスが付けられます。
a – 追加
d – 削除
u – 更新

プレフィックスの後に感嘆符（!）が付いている場合、そのリソースが「上書きして更新」によって処理されたことを意味します。

-c <コンフィグレーション名> / -r <リリース名>
変更を取得するリリースとコンフィグレーションを指定します。デフォルトでは、ローカルリソースに割り当てられている作業コンフィグレーションが使用されます。

<ファイル名/フォルダー名>
リポジトリから更新する 1 つ以上のローカルファイルまたはフォルダーの名前。ファイル名とフォルダー名は混在させることができ、絶対パス名または相対パス名を使用して指定します。省略すると、現在の作業フォルダーが使用されます。

例

リポジトリの最新の変更内容で、サブフォルダー「docum」を更新します。

```
C:¥test¥projects¥Project B>escm update docum
```

2.5.10 log – バージョンヒストリの表示 [SCM、DMS]

ローカルでチェックアウトしたリソースやドキュメント、または SCM や DMS リポジトリ内のリソースまたはドキュメントについて、バージョンヒストリを表示します。

コマンド構文：

```
escm log [-s <サーバ名> -d <リポジトリ名> -po <ポート>
      -p <プロジェクト名> -r <リリース名> -c <コンフィグレーション名>]
      <ファイル名/フォルダー名> [<ファイル名/フォルダー名> ...]
```

<ファイル名/フォルダー名>
バージョンヒストリを表示する 1 つ以上のファイルまたはフォルダーの名前。<ファイル名/フォルダー名>で指定したファイルとフォルダーがローカルリソースではない場合、以下のオプションを使用して場所を指定します。

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
リソースが含まれているリポジトリの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-p <プロジェクト名>
リソースが含まれているプロジェクトまたはライブラリの名前。

-r <リリース名>
ファイル/フォルダーが存在するリリースの名前。

-c <コンフィグレーション名>
リソースが含まれているコンフィグレーションの名前。

例

「ma_2.h」というローカルファイルのバージョンヒストリを表示します。

```
C:¥test¥projects¥Project B>escm log "source¥Module A¥ma_2.h"
```

「Sample」リポジトリ内にある「ma_2.h」というファイルのバージョンヒストリを表示します。

```
C:¥test>escm log -s estenabler1 -d Sample -p "Project B" -r 1.0 -c MAIN
"source¥Module A¥ma_2.h"
```

2.5.11 getversion – ヒストリからのバージョンの復元 [SCM]

ローカルファイルのバージョンを、リポジトリ内のヒストリのバージョンに置き換えます。

指定したファイルの内容とバージョン情報が、指定したヒストリバージョンの内容とバージョン情報に置換されます。

コマンド構文：

```
escm {getversion | getver} -v <バージョン> <ファイル名>
```

-v <バージョン>
復元するバージョンの数値。

<ファイル名>
復元するバージョンに置き換えるローカルファイルのパス名。

例

ローカルファイル「docum¥A1.txt」の内容をバージョン「3」に置き換えます。

```
C:¥test¥projects¥Project B>escm getversion -v 3 docum¥A1.txt
```

2.5.12 markmerged – マージ済みリソースのマーク [SCM]

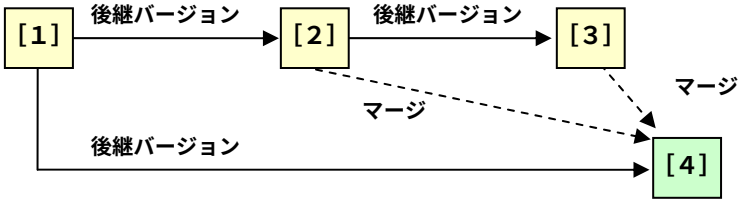
ファイルのローカルバージョンを、マージ済みバージョンとしてマークします。これは、このローカルバージョンのローカルワークスペースファイルの内容が、リポジトリにコミットされた最新バージョンの内容とマージされていることを意味します。マージを実行するユーザーは、内容が正しくマージされているか確認する必要があります。

マージ済みリソースをマークすると、リソースのローカルメタ情報が更新されます。リポジトリに対する書き込みは実行されません。ローカルリソースを通常どおりにリポジトリにコミットすると、マージ済み情報を保存できます。

例

1. ユーザーA とユーザーB が、「Test.txt [1]」という同一ファイルをチェックアウトします。
2. ユーザーA がリポジトリに変更をコミットし、「Test.txt [2]」を生成します。

3. ユーザーB もまた、ファイルを変更しています。したがって、ユーザーB は、ユーザーA が行った変更をユーザーB のローカルバージョンにマージする必要があります。その後、ユーザーB は、ローカルバージョン上で markmerged コマンドを呼び出しますが、リポジトリにはコミットしません。
4. ユーザーA がファイルを変更し、再びコミットして、「Test.txt [3]」を生成します。
5. ユーザーB は、この 2 回目の変更に気づき、再び変更内容をマージして、ユーザーB のローカルバージョンをマージ済みとしてマークします。ここで、ユーザーB はローカルバージョンをリポジトリにコミットし、マージ済みの「Test.txt [4]」を生成します。
6. リポジトリには、「Test.txt」に関する以下の情報が保存されています。



コマンド構文：

```
escm {markmerged | mm} [-ct] <ファイル名> [<ファイル名> ...]
```

-ct
　ファイルのタイムスタンプの変更をリソースの変更として扱います。これにより、ファイルの内容が変更されていなくても、更新日時が変更されていればそのファイルは変更されているとみなされ、リポジトリに新しいバージョンとしてコミットできるようになります。このオプションは、ファイルの更新日時を取り扱うアプリケーションの場合に役立ちます。

<ファイル名>
　マージ済みとしてマークする 1 つ以上のローカルファイルの名前。

例

ファイルをマージ済みとしてマークします。

```
C:¥test¥projects¥Project B¥bin¥debug>escm markmerged shop.dta
```

2.5.13 importcommit – リポジトリ内のリソースの作成 [SCM]

importcommit コマンドは、ローカルワークスペースを使用しない「上書きしてコミット」として動作します。ファイルをコミットするには、「-f」オプションを使用します。「-f」オプションを省略すると、リポジトリにフォルダーが作成されます。「-i」オプションを使用すると、リポジトリ内の親フォルダーを指定できます。「-i」オプションを省略すると、リソースはプロジェクトのルートフォルダーにコミットされます。作成するアイテムは、「-n」オプションで名前を指定する必要があります。

コマンド構文：

```
escm {importcommit | ici} -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -r <リリース名> -c <コンフィグレーション名>
      -m <コメント> -n <リソース名> [-po <ポート> -i <アイテムパス>
      -t <タスク [;タスク;...]> -f <ファイル>]
```

`-s <サーバ名>`
リポジトリが存在するサーバの名前。

`-d <リポジトリ名>`
リソースをコミットするリポジトリの名前。

`-p <プロジェクト名>`
リソースをコミットするプロジェクトの名前。

`-r <リリース名>`
リソースをコミットするリリースの名前。

`-c <コンフィグレーション名>`
リソースをコミットするコンフィグレーションの名前。

`-m <コメント>`
新しいリソースバージョンと一緒に保存する短い説明（コミットコメント）。

`-n <リソース名>`
リポジトリ内で作成または更新（バージョン付け）するリソースの名前。

`-po <ポート>`
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、`9500` です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

`-i <アイテムパス>`
リソースをコミットするリポジトリ内の親フォルダーのパス名。

`-t <タスク[;タスク;...]>`
変更をコミットする 1 つ以上のタスクの ID を、セミコロンで区切って指定します。ID の前部にパディングされている「0」は、省略できます。

`-f <ファイル>`
内容をリポジトリにコミットするローカルファイルのパス名。指定しないと、フォルダーが作成されます。

例

リポジトリ内の「ma_2.h」ファイルの新しいバージョンとして、「Contents.txt」ファイルの内容をコミットします。

```
C:¥>escm importcommit -s estescmtstxp -d Sample -p "Project B" -r 1.0
-c MAIN -i "¥source¥Module A¥" –n ma_2.h -f C:¥Tmp¥Contents.txt
-m "Imported from tmp"
```

リポジトリ内の既存のフォルダー「Module A」に、新しいサブフォルダー「Test」を作成します。

```
C:¥ >escm importcommit -s estescmtstxp -d Sample -p "Project B" -r 1.0
-c MAIN -i "¥source¥Module A" -n Test -m "Created new test folder"
```

2.5.14 compare – ローカルリソースとリポジトリの比較 [SCM、DMS]

注意: このコマンドは、＜ファイル名/フォルダー名＞の引数を指定しない場合、現在の作業フォルダーに適用されます。

ローカルリソースを、リポジトリ内の指定した SCM 作業コンフィグレーションまたは DMS ライブラリのファイルおよびフォルダーと比較し、結果を表示します。

差分が検出された各リソースについて、1 行の情報が出力されます。変更されていないリソースについては、出力されません。出力行は、変更マーカーで始まります。変更マーカーの値は、以下のとおりです。

u - リソースがローカル環境で更新されました。

U – リソースがリポジトリ内で更新されました。

d – リソースがローカル環境で削除されました。

D – リソースがリポジトリから削除されました。

a –リソースが登録されましたが作業コンフィグレーションの一部ではありません。

A – リソースがリポジトリに追加されました。

? - リソースがローカル環境に追加されましたが登録されていません。

これらのマーカーは、組み合わせて表示されることもあります。

変更マーカーの後に、リソースの種類（ファイルまたはフォルダー）が表示されます。リソースがファイルの場合、ローカル変更が行われたベースバージョン、すなわちチェックアウトされたバージョンが表示されます。

コマンド構文：

```
escm {compare | cmp} [-r -ct] [<ファイル名/フォルダー名> ...]
```

-r
　指定したフォルダーの下位構造を処理します。

-ct
　ファイルのタイムスタンプの変更をリソースの変更として扱います。これにより、ファイルの内容が変更されていなくても、更新日時が変更されていればそのファイルは変更されているとみなされ、リポジトリに新しいバージョンとしてコミットできるようになります。このオプションは、ファイルの更新日時を取り扱うアプリケーションの場合に役立ちます。

<ファイル名/フォルダー名>
　リポジトリと比較する 1 つ以上のローカルファイルまたはフォルダーの名前。ファイル名とフォルダー名は混在させることができ、絶対パス名または相対パス名を使用して指定します。省略すると、現在の作業フォルダーが使用されます。

例

ローカルサブフォルダー「source」とその下位構造を、作業コンフィグレーションと比較します。

```
C:¥test¥projects¥Project B>escm compare -r source
```

2.5.15 lstatus – リソースのローカルステータスの表示 [SCM、DMS]

このコマンドは、＜ファイル名/フォルダー名＞の引数を指定しない場合、現在の作業フォルダーに適用されます。

各リソースについて、変更マーカーで開始される 1 行の情報が表示されます。変更マーカーの値は、以下のとおりです。

u - リソースがローカル環境で更新されました。

d – リソースがローカル環境で削除されました。

? - リソースがローカル環境で追加されましたがリポジトリに登録されていません。

マーカーが表示されていない場合、リソースは最後のチェックアウトから変更されていません。変更マーカーの後に、リソースの種類（ファイルまたはフォルダー）が表示されます。リソースがファイルの場合、ローカル変更が行われたベースバージョン、すなわちチェックアウトされたバージョンが表示されます。

コマンド構文：

```
escm {lstatus | lst} [<ファイル名/フォルダー名> ...]
```

<ファイル名/フォルダー名>
ステータスを表示する 1 つ以上のローカルファイルまたはフォルダーの名前。ファイル名とフォルダー名は混在させることができ、絶対パス名または相対パス名を使用して指定します。省略すると、現在の作業フォルダーが使用されます。

例

ローカルサブフォルダー「Module A」のステータスを表示します。

```
C:¥test¥projects¥Project B¥source¥Module A>escm lstatus
```

2.5.16 diff – ファイル内容の差分の表示 [SCM、DMS]

指定したファイルの 2 つのバージョン間の差分を出力します（このコマンドは、フォルダーに対しては行えません）。1 つ以上のファイルを引数として指定できます。コマンドオプションを指定しない場合、（割り当てられている作業コンフィグレーションに応じて）ローカルワークスペースのファイルバージョンと、リポジトリ内の最新の表示可能バージョンが比較されます。

以下のどれかの比較モードを指定できます。

-b： ローカルバージョンと、チェックアウトしたリポジトリのバージョン（ベースバージョン）を比較します。

-v: ローカルバージョンと、リポジトリ内の指定したバージョンを比較します。

-h / -i: ローカルバージョンと、指定した SCM コンフィグレーションまたは DMS ライブラリ内で現在表示可能なリポジトリバージョンを比較します。

ファイル内の差分が検出された行が、変更内容を示すプレフィックスと一緒に出力されます。

+ - 行が追加されている場合

- - 行が削除されている場合

c – 行が変更されている場合

! - ローカルバージョンとリポジトリバージョンに矛盾する変更がある場合

コマンド構文：

```
escm diff [-b | -v <バージョン> | -h <リリース名>
      -i <コンフィグレーション名>] <ファイル名> [<ファイル名> ...]
```

-b
ローカルファイルを、そのファイルをチェックアウトしたリポジトリのバージョン（ベースバージョン）と比較します。

-v <バージョン>
ローカルファイルと、指定したリポジトリのバージョンを比較します。

-h <リリース名> / -i <コンフィグレーション名>
ローカルファイルと、指定したコンフィグレーション内の現在表示可能なリポジトリバージョンを比較します。「-i」と「-h」は、同時に指定する必要があります。

<ファイル名>
該当するリポジトリバージョンと比較する 1 つ以上のローカルファイルの名前。

例

ローカルファイルと、リポジトリ内の最新バージョンを比較します。

```
C:¥test¥projects¥Project B¥source¥Module A>escm diff ma1.cpp
```

2.5.17 commitnotif – 通知対象の表示 [SCM、DMS]

指定したリソース、または SCM コンフィグレーションまたは DMS ライブラリ全体について、実行ユーザーによって登録されている通知対象を表示します。

通知対象は、SCM プロジェクトのリソースまたは DMS のフォルダーとドキュメントに対して登録できます。他のチームメンバーによってリソースの変更がコミットされると、登録ユーザーにメールで通知されます。

コマンド構文：

```
escm {commitnotif | cn} -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -r <リリース名> -c <コンフィグレーション名>
      [-po <ポート> -i <アイテムパス>]
```

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
プロジェクトまたはライブラリが含まれているリポジトリの名前。

-p <プロジェクト名>
通知対象を表示するプロジェクトまたはライブラリの名前。

-r <リリース名>
通知対象を表示するリリースの名前。

-c <コンフィグレーション名>
通知対象を表示するコンフィグレーションの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-i <アイテムパス>
通知対象を表示するアイテムのパス名。省略すると、指定したコンフィグレーション内の実行ユーザーのすべての通知対象が表示されます。

例

現在のユーザー「ESTSCMTSTXP¥Test」が、「source/Module A」について通知対象を登録しているかどうかを確認して、表示します。

```
C:¥>escm commitnotif -s estescmtstxp -d Sample -p "Project B" -r 1.0 -c
MAIN -i "/source/Module A"
```

コンフィグレーション内の現在のユーザー「ESTSCMTSTXP¥Test」のすべての登録済み通知対象を確認して、表示します。

```
C:¥>escm commitnotif -s estescmtstxp -d Sample -p "Project B" -r 1.0
-c MAIN
```

2.5.18 addcommitnotif – 通知対象の追加 [SCM、DMS]

指定した SCM プロジェクトのリソースまたは DMS のフォルダーまたはドキュメントに対して、メールによるコミット通知対象を登録します。

リソースがフォルダーの場合、通知対象は、フォルダー内のすべてのサブフォルダーとファイルを含めたフォルダー全体に登録されます。

コマンド構文：

```
escm {addcommitnotif | acn} -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -r <リリース名> -c <コンフィグレーション名>
      -i <アイテムパス> [-po <ポート> -u <ユーザー名>]
```

`-s <サーバ名>`
リポジトリが存在するサーバの名前。

`-d <リポジトリ名>`
プロジェクトまたはライブラリが含まれているリポジトリの名前。

`-p <プロジェクト名>`
アイテムが含まれているプロジェクトまたはライブラリの名前。

`-r <リリース名>`
アイテムを含むリリースの名前。

`-c <コンフィグレーション名>`
アイテムを含むコンフィグレーションの名前。

`-i <アイテムパス>`
通知対象を登録するアイテムのパス名。

`-po <ポート>`
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

`-u <ユーザー名>`
通知対象として登録するユーザーの名前。ADM ユーザーの ID、または任意のメールアドレス（メーリングリストなど）を指定できます。メールアドレスを指定する場合、指定するメールアドレスを角括弧（[ ]）で囲む必要があります。省略すると、実行ユーザーが指定されます。

例

フォルダー「/Documents/QA」にコミット通知対象を登録します。

```
C:¥>escm addcommitnotif -s estenabler1 -d Sample -p Demo -r 5.00 -c MAIN
   -i /Documents/QA
```

「/Documents/ReadMe.txt」ファイルに、コミット通知対象としてメーリングリスト「mailing_grp」を登録します。

```
C:¥>escm addcommitnotif -s estenabler1 -d Sample -p Demo -r 5.00 -c MAIN
   -i /Documents/ReadMe.txt –u [mailing_grp].
```

2.5.19 delcommitnotif – 通知対象の削除 [SCM、DMS]

指定したリソース、または SCM コンフィグレーションや DMS ライブラリ全体から、ユーザーの通知対象を削除します。

コマンド構文：

```
escm {delcommitnotif | dcn} -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -r <リリース名> -c <コンフィグレーション名>
      [-po <ポート> -i <アイテムパス> -u <ユーザー名>]
```

`-s <サーバ名>`
リポジトリが存在するサーバの名前。

`-d <リポジトリ名>`
プロジェクトまたはライブラリが含まれているリポジトリの名前。

`-p <プロジェクト名>`
通知対象の登録を削除するプロジェクトまたはライブラリの名前。

`-r <リリース名>`
通知対象の登録を削除するリリースの名前。

`-c <コンフィグレーション名>`
通知対象の登録を削除するコンフィグレーションの名前。

`-po <ポート>`
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、`9500` です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

`-i <アイテムパス>`
通知対象の登録を削除するアイテムのパス名。省略すると、指定したコンフィグレーション内のすべての通知対象が削除されます。

`-u <ユーザー名>`
通知対象の登録を削除するユーザーの名前。ADM ユーザーの ID、または任意のメールアドレス（メーリングリストなど）を指定できます。メールアドレスを指定する場合、指定するメールアドレスを角括弧（[ ]）で囲む必要があります。省略すると、実行ユーザーが指定されます。

例

コンフィグレーション「MAIN」から、現在のユーザーのすべての通知対象を削除します。

```
C:¥>escm delcommitnotif -s estescmtstxp -d Sample -p "Project B" -r 1.0
-c MAIN
```

フォルダー「/docs」から、メーリングリスト「mailing_grp」の通知対象を削除します。

```
C:¥>escm delcommitnotif -s estescmtstxp -d Sample -p "Project B" -r 1.0
-c MAIN –i /docs –u [mailing_grp]
```

2.5.20 resourcelocks – リソースロックの管理 [SCM、DMS]

SCM コンフィグレーションまたは DMS ライブラリ内の指定したリソースまたはすべてのリソースに設定されているロックを表示します。親フォルダーがロックされたことによりリソースが暗黙のうちにロックされている場合、親フォルダーのロックだけが表示されます。

SCM と DMS では、プロジェクトまたはライブラリのリソースを排他的にロックすることができます。リソースをロックしたユーザーだけが、そのリソースへの変更をコミットできます。この機能は、複数の開発者が同じリソース上で作業を行う分散型開発環境の場合、特に効果的です。リポジトリに変更をコミットするときに、リソースバージョンをマージする必要がないからです。DMS では、チェックアウトしたドキュメントは自動的にロックされ、ドキュメントがチェックインされるとロックが解除されます。

指定したリソースに設定されているすべてのロックを表示し、管理できます。指定した期間にロックされたリソースだけを選択し、その選択されたリソースのロックを解除することもできます。これにより、例えば、古いロックを解除できます。リソースをロックしたユーザーに対して、既存のリソースロックまたはリソースロック解除の情報を、メールで通知できます。

検出されたすべてのリソースロックについて、以下の情報が表示されます。

- ロックされているリソースのパス
- ロックを設定したユーザー
- ロックの作成日時
- ロック設定時にユーザーが入力したコメント

コマンド構文：

```
escm {resourcelocks | rl} [-a –mail -unlock] -s <サーバ名>
      -d <リポジトリ名> -p <プロジェクト名> -r <リリース名>
      -c <コンフィグレーション名> [-po <ポート> -i <アイテムパス>]
      [-ld <ロック作成日(yyyy-MM-dd)> | -lt <ロック日数>]
```

`-a`
: すべてのユーザーのロックを選択して表示します。省略すると、実行ユーザーが設定したリソースロックだけが対象になります。

`-mail`
: 指定したリソースをロックしたすべてのユーザーに対して、ロック通知メールを送信します。「`-mail`」オプションと「`–a`」オプションを併用すると、すべてのユーザーにメールを送信します。併用しない場合、実行ユーザーのみに送信します。「`-mail`」オプションと「`–unlock`」オプションを併用するとリソースのロック解除を通知するメールが送信されます。「`-ld`」または「`–lt`」オプションを指定すると、指定した期間内に設定されたリソースロックだけが対象になります。

`-unlock`
: 選択したリソースのロックを解除します。「`-mail`」オプションを指定すると、ユーザーにリソースのロック解除を通知するメールが送信されます。「`-ld`」または「`–lt`」オプションを指定すると、指定した期間内に設定されたリソースロックだけが対象になります。

`-s <サーバ名>`
: リポジトリが存在するサーバの名前。

`-d <リポジトリ名>`
: プロジェクトまたはライブラリが含まれているリポジトリの名前。

`-p <プロジェクト名>`
: ロック設定を表示するプロジェクトまたはライブラリの名前。

`-r <リリース名>`
: ロック設定を表示するリリースの名前。

`-c <コンフィグレーション名>`
: ロック設定を表示するコンフィグレーションの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-i <アイテムパス>
ロック設定を表示するアイテムのパス名。省略すると、指定したコンフィグレーション内のすべてのロックが表示されます。

-ld <ロック作成日(yyyy-MM-dd)>
リソースがロックされた日付。日付は、yyyy-MM-dd 形式で指定します。指定した日付以前にロックされたリソースだけが選択されます。「-ld」オプションと「-lt」オプションを同時に指定することはできません。

-lt <ロック日数>
リソースがロックされている日数。指定した日数よりも長くロックされているリソースだけが選択されます。「-lt」オプションと「-ld」オプションを同時に指定することはできません。

例

指定したコンフィグレーション内のすべてのロックを表示します。

```
C:¥>escm resourcelocks -s estescmtstxp -d Sample -p "eNet" -r 1.6
-c MAIN –a
```

2.5.21 setresourcelock – リソースのロック [SCM]

実行ユーザー用に、指定したリソースを排他的にロックします。

リソースがフォルダーの場合、ロックは、フォルダー内のすべてのサブフォルダーとファイルを含めたフォルダー全体に設定されます。

コマンド構文：

```
escm {setresourcelock | srl} -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -r <リリース名> -c <コンフィグレーション名>
      -i <アイテムパス> [-po <ポート> -m <コメント>]
```

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
プロジェクトが含まれているリポジトリの名前。

-p <プロジェクト名>
アイテムを含むプロジェクトの名前。

-r <リリース名>
アイテムを含むリリースの名前。

-c <コンフィグレーション名>
アイテムを含むコンフィグレーションの名前。

-i <アイテムパス>
ロックするアイテムのパス名。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-m <コメント>
ロックに関する短いコメント。

例

現在のユーザー用に、フォルダー「docum」とその下位構造を排他的にロックします。

```
C:¥>escm setresourcelock -s estescmtstxp -d Sample -p "Project B"
-r 1.0 -c MAIN -i ¥docum –m "Until 04/2012".
```

2.5.22 delresourcelock – リソースのロック解除 [SCM]

指定したリソースまたは SCM コンフィグレーション全体から、実行ユーザーが設定したリソースのロックを解除します。

コマンド構文：

```
escm {delresourcelock | drl} -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -r <リリース名> -c <コンフィグレーション名>
      [-po <ポート> -i <アイテムパス>]
```

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
プロジェクトが含まれているリポジトリの名前。

-p <プロジェクト名>
アイテムを含むプロジェクトの名前。

-r <リリース名>
アイテムを含むリリースの名前。

-c <コンフィグレーション名>
アイテムを含むコンフィグレーションの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-i <アイテムパス>
ロックを解除するアイテムのパス名。省略すると、指定したコンフィグレーション内にある実行ユーザーのすべてのロックが解除されます。

例

「MAIN」コンフィグレーションから、現在のユーザーが設定したすべてのリソースのロックを解除します。

```
C:¥>escm delresourcelock -s estescmtstxp -d Sample -p "Project B" -r 1.0
-c MAIN
```

2.5.23 movren – 移動済みまたは名前変更済みリソースのマーク [SCM]

注意: このコマンドは、＜ファイル名/フォルダー名＞の引数を指定しない場合、現在の作業フォルダーに適用されます。

バージョンヒストリを保存するために、移動または名前変更したリソースをマークします。

ローカルワークスペースでリソースを別の場所に移動したり、リソースの名前を変更したりした場合、リポジトリ内のリソースの完全なバージョンヒストリを保持するために、このコマンドを呼び出す必要があります。リソースの各バージョンはすべて異なる名前になり、バージョンヒストリに表示されます。

リソースの名前変更および移動はローカルワークスペース内で実行し、実行後にリポジトリにコミットする必要があります。名前変更や移動をするたびにリソースのコミットをすることを推奨します。そうでない場合、以下の例に示すような予期せぬ結果が起こる可能性があります。

1. 「Test.java」ファイルを「Test2.java」に名前変更し、「Test2.java」 ファイルを名前変更済みとしてマークします。

2. 新しい Java クラスを作成するために、テンプレートフォルダーから「Test.java」ファイルを元の場所にコピーします。その後、そのファイルを「Test3.java」に名前変更し、名前変更済みとしてマークします。

3. 「Test2.java」と「Test3.java」を両方ともコミットします。

コミットの後、「Test3.java」は新しいリソースとして追加されずに、「Test.java」ファイルの後継バージョンとしてリポジトリに追加されます。

コマンド構文：

```
escm movren -j <リソースパス> [<ファイル名/フォルダー名>]
```

-j <リソースパス>
　移動前または名前を変更する前のリソースのフルパス名。

<ファイル名/フォルダー名>
　移動後または名前を変更した後のファイルまたはフォルダーの新しいパス名。ファイル名やフォルダー名は、絶対パス名または相対パス名を使用して指定します。省略すると、現在の作業フォルダーが使用されます。

例

ファイルの名前を変更し、SCM に通知します。

```
C:¥test¥projects¥Project B¥docum>rename B.txt Bnew.txt
C:¥test¥projects¥Project B¥docum>escm movren -j B.txt Bnew.txt
```

2.5.24 upditem – アイテムプロパティの更新 [SCM、DMS]

SCM プロジェクトまたは DMS ライブラリ内の指定したリソースのプロパティまたはフラグを変更します。

コマンド構文：

```
escm upditem -s <サーバ名> -d <リポジトリ名> -p <プロジェクト名>
      -r <リリース名> -c <コンフィグレーション名> -i <アイテムパス>
      [-x <0/1(有効化)>] [-po <ポート> -excs <on/off>
      -prp <プロパティ;...>]
```

-s <サーバ名>
　リポジトリが存在するサーバの名前。

-d <リポジトリ名>
　プロジェクトが含まれているリポジトリの名前。

-p <プロジェクト名>
　アイテムを含むプロジェクトの名前。

`-r <リリース名>`
アイテムを含むリリースの名前。

`-c <コンフィグレーション名>`
アイテムを含むコンフィグレーションの名前。

`-i <アイテムパス>`
プロパティを変更するアイテムのパス名。

`-x <0/1(有効化)>`
値 "1" は、指定したファイルに対する実行ファイル属性を有効にします。値 "0" は、指定したファイルに対する実行ファイル属性を無効にします（デフォルト）。有効にすると、ファイルの更新やチェックアウトにより新しいファイルを生成した場合、実行ファイル属性が設定されます。ローカルワークスペースにすでに設定されている属性は新しいバージョンのファイルを更新やチェックインしても変わりません。新しいファイルをリポジトリに追加する場合、実行ファイル属性がローカルファイルに設定されます。新しいバージョンのファイルをコミットしても、既存バージョンの実行ファイル属性は変わりません。

実行ファイル属性は、UNIX/Linux システムで実行可能なファイルに対して必要です。この機能は、Java 6.0 環境でのみ適切に動作します。そのため、Java 6.0 が UNIX/Linux クライアントにインストールされている必要があります。この場合、サーバの Java のバージョンとオペレーティングシステムは、関係しません。

`-po <ポート>`
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、`9500` です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

`-excs <on/off>`
アイテムをコードスキャンから除外するかどうかを定義します（除外すると、ソースコードとして処理されなくなります）。フォルダーを指定した場合、この設定はフォルダー内のすべてのリソースに適用されます。

`-prp <プロパティ>`

このオプションは今版では利用できません。

例

フォルダーをソースコードスキャンから除外します。

```
C:¥>escm upditem -s estescmtstxp -d Sample -p "Project B" -r 1.0 -c MAIN
-i "source¥Module A" –excs off
```

2.5.25 mark – エクスポート用リソースのマーク [SCM]

export コマンド用にリソースをマークします。リソースがフォルダーの場合、この設定はフォルダー内のすべてのリソースに適用されます。

コンフィグレーションまたはリソース構造をエクスポートする場合、含まれているすべてのリソースをエクスポートするのか、またはエクスポート用に明示的にマークされたリソースだけをエクスポートするのかを選択できます（「`export` コマンド」を参照）。

コマンド構文：

```
escm mark [-m] -s <サーバ名> -d <リポジトリ名> -p <プロジェクト名>
      -r <リリース名> -c <コンフィグレーション名> -i <アイテムパス>
      [-po <ポート>]
```

-m
エクスポート用マークを削除します。

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
プロジェクトが含まれているリポジトリの名前。

-p <プロジェクト名>
アイテムを含むプロジェクトの名前。

-r <リリース名>
アイテムを含むリリースの名前。

-c <コンフィグレーション名>
アイテムを含むコンフィグレーションの名前。

-i <アイテムパス>
エクスポート用にマークするアイテムのパス名。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

例

サブフォルダー「bin」をエクスポート用にマークします。

```
C:¥>escm mark -s estescmtstxp -d Sample -p "Project B" -r 1.0 -c MAIN
-i bin.
```

2.5.26 accessrights – アクセス権限の表示 [SCM、DMS]

SCM プロジェクトまたは DMS ライブラリのフォルダーまたはファイルのアクセス権限を表示します。

コマンド構文：

```
escm {accessrights | ar} -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -r <リリース名> -c <コンフィグレーション名>
      -i <アイテムパス> [-po <ポート>]
```

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
プロジェクトまたはライブラリが含まれているリポジトリの名前。

-p <プロジェクト名>
アイテムが含まれているプロジェクトまたはライブラリの名前。

-r <リリース名>
アイテムを含むリリースの名前。

-c <コンフィグレーション名>
アイテムを含むコンフィグレーションの名前。

-i <アイテムパス>
アクセス権限を表示するアイテムのパス名。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

例

「/Development/A.txt」というファイルのアクセス権限を表示します。

```
C:¥>escm accessrights -s estenabler1 -d Sample -p Demo -r 5.00 -c MAIN
   -i /Development/A.txt
```

2.5.27 setaccessrights – アクセス権限の定義 [SCM、DMS]

SCM プロジェクトまたは DMS ライブラリのアイテム（フォルダーまたはファイル）のアクセス権限を定義します。「-e」オプション、「-w」オプション、または「-o」オプションをいずれも指定しない場合、指定したユーザーまたはグループに読み取り権限が定義されます。

オーナー権限の代わりに読み取り権限を定義するなど、ユーザーのアクセス権限を限定する場合には、delaccessrights コマンドを利用して、そのユーザーからアクセス権限を削除しておく必要があります。アクセス権限を拡張する場合には、事前にアクセス権限を削除する必要はありません。

コマンド構文：

```
escm {setaccessrights | sar} [-a] [-e | -w | -o] -s <サーバ名>
      -d <リポジトリ名> -p <プロジェクト名> -r <リリース名>
      -c <コンフィグレーション名> -i <アイテムパス> [-po <ポート>]
      {-g <グループ名> | -u <ユーザー名>}
```

-a
指定したアイテム（フォルダー）の下位構造にも定義を適用します。

-e
アイテムに対するデフォルトの権限（すべてのアクセスグループメンバーのオーナー権限）を設定します。

-w
指定したユーザーまたはグループに書き込み権限を定義します。この設定により、ユーザーまたはグループはアイテムを変更できるようになります。アイテムがフォルダーの場合は、フォルダー内に新しいリソースを作成できます。アイテムの削除またはアクセス権限の変更は許可されません。

-o
指定したユーザーまたはグループにオーナー権限を定義します。この設定により、ユーザーまたはグループは、アイテムの削除、およびアイテムのアクセス権限の変更ができるようになります。

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
プロジェクトまたはライブラリが含まれているリポジトリの名前。

-p <プロジェクト名>
アイテムが含まれているプロジェクトまたはライブラリの名前。

-r <リリース名>
アイテムを含むリリースの名前。

-c <コンフィグレーション名>
　アイテムを含むコンフィグレーションの名前。

-i <アイテムパス>
　アクセス権限を変更するアイテムのパス名。

-po <ポート>
　リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-g <グループ名>
　アクセス権限を割り当てるグループの ID。

-u <ユーザー名>
　アクセス権限を割り当てるユーザーの ID。

例

フォルダー「common」とその下位構造に、ユーザー「ESTESCMTSTXP¥User1」の書き込み権限を設定します。

```
C:¥>escm setaccessrights -s estescmtstxp -d Sample -p "Project B" -r 1.0
-c MAIN -i ¥source¥common -u estescmtstxp¥User1 -w –a
```

フォルダー「source」とその下位構造にデフォルトのアクセス権限を設定します。

```
C:¥>escm setaccessrights -s estescmtstxp -d Sample -p "Project B" -r 1.0
-c MAIN -i ¥source -e –w
```

2.5.28 delaccessrights – アクセス権限の削除 [SCM、DMS]

SCM プロジェクトまたは DMS ライブラリのアイテム（フォルダーまたはファイル）に定義されているアクセス権限から、ユーザーまたはグループを削除します。

オーナー権限の代わりに読み取り権限を定義するなど、setaccessrights コマンドを使用して、アクセス権限を限定する場合には、そのユーザーからアクセス権限を削除しておく必要があります。アクセス権限を拡張する場合には、事前にアクセス権限を削除する必要はありません。

コマンド構文：

```
escm {delaccessrights | dar} [-a] -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -r <リリース名> -c <コンフィグレーション名>
      -i <アイテムパス> [-po <ポート>]
      {-g <グループ名> | -u <ユーザー名>}
```

-a
　指定したアイテム（フォルダー）の下位構造にも変更を適用します。

-s <サーバ名>
　リポジトリが存在するサーバの名前。

-d <リポジトリ名>
　プロジェクトまたはライブラリが含まれているリポジトリの名前。

-p <プロジェクト名>
　アイテムが含まれているプロジェクトまたはライブラリの名前。

-r <リリース名>
　アイテムを含むリリースの名前。

-c <コンフィグレーション名>
アイテムを含むコンフィグレーションの名前。

-i <アイテムパス>
アクセス権限を変更するアイテムのパス名。

-g <グループ名>
アクセス権限を削除するグループの ID。

-u <ユーザー名>
アクセス権限を削除するユーザーの ID。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

例

フォルダー「common」とその下位構造から、ユーザー「ESTESCMTSTXP¥User2」のすべてのアクセス権限を削除します。

```
C:¥>escm delaccessrights -s estescmtstxp -d Sample -p "Project B" -r 1.0
-c MAIN -i ¥source¥common -u estescmtstxp¥User2 –a
```

2.5.29 edit – ローカルリソースの編集状態の切替え/表示 [SCM]

注意: このコマンドは、＜ファイル名＞の引数を指定しない場合、現在の作業フォルダーに適用されます。

指定したローカルファイルの編集状態を設定または削除します。このアクションを呼び出すことができるのは、ローカルリソース変更の監視が有効になっている場合だけです（詳細は、「monitor コマンド」を参照）。ADM クライアントインターフェース（Eclipse）では、編集状態が自動的に設定および削除されます。Eclipse を使用していない場合は、このコマンドを使用して、編集状態を手作業で設定または削除できます。

monitor コマンドの「sync」オプションを使用すると、1 回の操作で、ローカルプロジェクトのすべてのファイルの編集状態を検索して、保存できます。

コマンド構文：

```
escm edit [{-u | –check | -list}] [<ファイル名> ...]
```

-u
指定したファイルから編集状態を削除します（未編集状態が設定されます）。

-check
リポジトリ内で、指定したファイルの 1 つが他のユーザーによって編集されているかどうかを確認します。他のユーザーによって編集されていない場合、すべてのファイルが編集済みとしてマークされます。それ以外の場合は、編集状態は設定されず、警告メッセージが表示されます。

-list
ローカルワークスペース内で指定したリソースを最後に編集したユーザーのリストを表示します。

<ファイル名>
表示する、または編集済み（「–u 」オプションを指定した場合には未編集）としてマークする 1 つ以上のローカルファイルの名前。ファイル名は、絶対パス名または相対パス名を使用して指定します。省略すると、現在の作業フォルダーのファイルが使用されます。

例

「FileA1.txt」ファイルと「FileA2.txt」ファイルを指定し、編集済みとしてマークします。

```
C:¥>escm edit "C:/test/work/Project A/FileA1.txt" "C:/test/work/Project
A/FileA2.txt"
```

フォルダー「/source」内のすべての編集済みリソースを表示します。

```
C:¥>escm edit -list "C:/test/work/Project A/source"
```

2.6 コンフィグレーションの管理

SCM プロジェクト内のコンフィグレーションを管理するには、以下のコマンドを使用します。

2.6.1 rtag – スナップショットまたはブランチの作成 [SCM]

リポジトリ内の指定した作業コンフィグレーションのすべてのコミット済みリソース（フォルダーおよびファイルバージョン）を含む、保護されたコンフィグレーション（スナップショット）を作成します。一般的に、スナップショットは、ビルド、テスト、または修正などの目的で、リリース内のプロジェクトの特定の状態をアーカイブするときに使用します。変更管理への自動反映が設定されている場合、対象のコンフィグレーションへのリンクが LCM 側に作成されます。

保護されたコンフィグレーション上でこのコマンドを呼び出すと、指定したオプションに応じて、保護されたコンフィグレーションの新しいブランチ（作業コンフィグレーション）またはコピーが作成されます。

命名規則

各種のクライアントオペレーティングシステムを適正にサポートするために、スナップショットの名前、ブランチの名前には以下の文字と同様、すべての制御文字を使用できません。
`* : ¥ / ? < > | "`
スナップショット名、ブランチ名にこれらの文字が含まれていると、スナップショットまたはブランチの作成に失敗します。

コマンド構文：

```
escm rtag [-snap | –branch] [-nolcm] -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -r <リリース名> -c <コンフィグレーション名>
      -n <スナップショット/ブランチ名> [-po <ポート> -m <コメント>
      -z <ステート名>]
```

`-snap`
　指定したコンフィグレーションの種類およびステータスに関係なく、スナップショットを作成します。このオプションは、既存スナップショットのコピーを作成するときに使用できます。

`-branch`
　ブランチを作成します。

`-nolcm`
　LCM への接続を無効にします。この場合、LCM の関連するコンフィグレーションリンクは作成されません。「変更管理への自動反映」がリポジトリで有効になっている場合は（`chmansett` コマンドを参照）、このオプションで接続を無効にできません。

`-s <サーバ名>`
　リポジトリが存在するサーバの名前。

`-d <リポジトリ名>`
　プロジェクトが含まれているリポジトリの名前。

`-p <プロジェクト名>`
　コンフィグレーションを格納しているプロジェクトの名前。

`-r <リリース名>`
　コンフィグレーションを格納しているリリースの名前。

-c <コンフィグレーション名>
スナップショットまたはブランチを作成するコンフィグレーションの名前。

-n <スナップショット/ブランチ名>
新しいスナップショットまたはブランチの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-m <コメント>
新しいコンフィグレーションの短い説明（スナップショットのみ）。

-z <ステート名>
新しいコンフィグレーションに割り当てるステートの名前（スナップショットのみ）。プロジェクトで使用できるコンフィグレーションステートのリスト（ファイル）を取得するには、exportstates コマンドを使用します。

例

「MAIN」コンフィグレーションの新しいスナップショットを作成します。

```
C:¥>escm rtag -n "Build 1.2.3" -s estescmtstxp -d Sample -p "Project B"
-r 1.0 -c MAIN
```

スナップショット「Build 1.2.3」に基づいて、新しいブランチを作成します。

```
C:¥>escm rtag -n MAIN_Japan -s estescmtstxp -d Sample -p "Project B" -r
1.0 -c "Build 1.2.3"
```

既存のスナップショット「Build 1.2.3」のコピーを作成します。

```
C:¥>escm rtag –snap -n "SnapCopy" -s estescmtstxp -d Sample
-p "Project B" -r 1.0 -c "Build 1.2.3" –m "Created for QA"
```

2.6.2 delconf – コンフィグレーションの削除 [SCM]

指定した SCM コンフィグレーションを削除します。変更管理への自動反映が設定されている場合、対象のコンフィグレーションへのリンクが LCM 側で削除されます。

注意：このアクション（削除）は元に戻すことができません。

コマンド構文：

```
escm delconf [-f -nolcm] -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -r <リリース名> -c <コンフィグレーション名>
      [-po <ポート>]
```

-f
確認を要求せずに、コンフィグレーションを削除します。

-nolcm
LCM への接続を無効にします。この場合、LCM の関連するコンフィグレーションリンクは削除されません。「変更管理への自動反映」がリポジトリで有効になっている場合は（chmansett コマンドを参照）、このオプションで接続を無効にできません。

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
プロジェクトが含まれているリポジトリの名前。

-p <プロジェクト名>
コンフィグレーションを格納しているプロジェクトの名前。

-r <リリース名>
コンフィグレーションを格納しているリリースの名前。

-c <コンフィグレーション名>
削除するコンフィグレーションの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

例

スナップショット「Build_1.2.3」を削除します。

```
C:¥>escm delconf -s estescmtstxp -d Sample -p "Project B" -r 1.0 -c
"Build_1.2.3"
```

2.6.3 renconf – コンフィグレーションの名前変更 [SCM]

指定した SCM コンフィグレーションの名前を変更します。変更管理が有効で、SCM リポジトリ用に設定されている場合、コンフィグレーション名の変更により、LCM 内のコンフィグレーションへのリンクもすべて更新されます。

コンフィグレーション名の制限

各種のクライアントオペレーティングシステムを適正にサポートするために、コンフィグレーションの名前には以下の文字と同様、すべての制御文字を使用できません。
* : ¥ / ? < > | "
コンフィグレーション名にこれらの文字が含まれていると、コンフィグレーションの名前変更に失敗します。

コマンド構文：

```
escm renconf [-nolcm] -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -r <リリース名> -c <コンフィグレーション名>
      [-po <ポート>] <新しいコンフィグレーション名>
```

-nolcm
LCM への接続を無効にします。この場合、LCM のコンフィグレーションリンクは更新されません。「変更管理への自動反映」がリポジトリで有効になっている場合は（chmansett コマンドを参照）、このオプションで接続を無効にできません。

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
プロジェクトが含まれているリポジトリの名前。

-p <プロジェクト名>
コンフィグレーションを格納しているプロジェクトの名前。

-r <リリース名>
コンフィグレーションを格納しているリリースの名前。

-c <コンフィグレーション名>
名前を変更するコンフィグレーションの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

<新しいコンフィグレーション名>
コンフィグレーションの新しい名前。

例

スナップショット「Build 1」の名前を「Build 1_pre」に変更します。

```
C:¥>escm renconf -s estescmtstxp -d Sample -p "Project B" -r 1.0 -c
"Build 1" "Build 1_pre"
```

2.6.4 closeconf – 作業コンフィグレーションのクローズ [SCM]

コンフィグレーションをクローズすると、その内容が保護されます。この操作は、作業用のブランチがリリースされた後、以降の変更を許可しない場合に役立ちます。変更管理が有効で、SCM リポジトリ用に設定されている場合、LCM の関連コンフィグレーションのリンクも更新されます。

コンフィグレーション名の制限

各種のクライアントオペレーティングシステムを適正にサポートするために、コンフィグレーションの名前には以下の文字と同様、すべての制御文字を使用できません。
* : ¥ / ? < > | "
コンフィグレーション名にこれらの文字が含まれていると、作業コンフィグレーションのクローズに失敗します。

コマンド構文：

```
escm closeconf [-nolcm] -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -r <リリース名> -c <コンフィグレーション名>
      [-po <ポート> -n <新しいコンフィグレーション名> -m <コメント>
      -z <ステート名>]
```

-nolcm
LCM への接続を無効にします。この場合、LCM の関連するコンフィグレーションリンクは更新されません。「変更管理への自動反映」がリポジトリで有効になっている場合は（chmansett コマンドを参照）、このオプションで接続を無効にできません。

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
プロジェクトが含まれているリポジトリの名前。

-p <プロジェクト名>
コンフィグレーションを格納しているプロジェクトの名前。

-r <リリース名>
コンフィグレーションを格納しているリリースの名前。

-c <コンフィグレーション名>
クローズするコンフィグレーションの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-n <新しいコンフィグレーション名>
コンフィグレーションの新しい名前。

-m <コメント>
コンフィグレーションの短い説明。

-z <ステート名>
コンフィグレーションに割り当てるステートの名前。プロジェクトで使用できるコンフィグレーションステートのリスト（ファイル）を取得するには、exportstates コマンドを使用します。

例

ブランチ「MAIN」をクローズして、「Build 1」という名前に変更します。

```
C:¥>escm closeconf -s estescmtstxp -d Sample -p "Project B" -r 1.0
-c MAIN -n "Build 1"
```

2.6.5 dellostfound – Lost & Foundフォルダーを空にする [SCM, DMS]

SCM プロジェクトまたは DMS ライブラリの Lost & Found フォルダーに含まれているすべての内容を削除します。

Lost & Found フォルダーには、削除されて、プロジェクトのコンフィグレーションまたはライブラリのエディションから参照されないすべてのリソースが含まれています。

注意：このアクション（削除）は元に戻すことができません。

コマンド構文：

```
escm dellostfound [-f] -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> [-po <ポート>]
```

-f
確認を要求せずに、Lost & Found フォルダーの内容を削除します。

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
プロジェクトまたはライブラリが含まれているリポジトリの名前。

-p <プロジェクト名>
プロジェクトまたはライブラリの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

例

Lost & Found フォルダー内のすべてのアイテムを削除します。

```
C:¥>escm dellostfound -s estescmtstxp -d Sample -p "Project B"
```

2.6.6 delintvers – 中間バージョンの削除 [SCM、DMS]

SCM プロジェクトまたは DMS ライブラリ内の指定したリリース、またはその先行リリース内で作成されたすべての中間ファイルバージョンの内容を削除します。

リソースの開発段階では、最終バージョンがリリースされるまでに、多数のバージョンがリポジトリにコミットされます。開発プロセスの間に、複数のリリースおよびコンフィグレーション（スナップショットまたはブランチ）が作成され、開発されたリソースのバージョンがコンフィグレーションに関連付けられます。

リリースの開発中は、各リソースのすべてのバージョンが必要です（例えば、予備または比較用として使用されます）。しかし、コンフィグレーションのリリース後は、スナップショットまたはブランチに対応したバージョン、および作業コンフィグレーションの最新のバージョンだけが有用なバージョンとなり、その他のバージョンは不要になります。

このコマンドを使用すると、最終リリース済みのコンフィグレーションの開発中に作成された、すべての中間バージョンの内容が物理的に削除されます。これにより、空きディスク領域を増やし、管理するファイルバージョン数を削減できます。

ファイルの内容は削除されますが、バージョン情報（作成者、コメント、コミットの日時など）は、バージョンヒストリとして保持されます。削除した中間バージョンに割り当てられていたタスクは、（ファイルの変更があった）次の後継バージョンにマージされます。

注意：このアクション（削除）は元に戻すことができません。

コマンド構文：

```
escm delintvers -s <サーバ名> -d <リポジトリ名> -p <プロジェクト名>
      -r <リリース名> [-po <ポート>]
```

`-s <サーバ名>`
リポジトリが存在するサーバの名前。

`-d <リポジトリ名>`
プロジェクトまたはライブラリが含まれているリポジトリの名前。

`-p <プロジェクト名>`
プロジェクトまたはライブラリの名前。

`-r <リリース名>`
中間バージョンの削除を開始するリリースの名前。

`-po <ポート>`
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、`9500` です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

例

リリース「2.0」以前のリリースで作成されたすべての中間バージョンを削除します。

```
C:¥>escm delintvers -s estescmtstxp -d Sample -p "Project B" -r 2.0
```

2.6.7 rdiff – コンフィグレーションの比較 [SCM]

注意: このコマンドは、サーバ、リポジトリ、およびプロジェクトの接続情報を検索できるように、ローカルワークスペースのフォルダー内から呼び出す必要があります。

2 つのコンフィグレーションのリソースを相互に比較します。

1 つのリリースおよびコンフィグレーション名だけを指定すると、そのコンフィグレーションのリソースが、ローカルワークスペースに割り当てられている作業コンフィグレーションと比較されます。2 つのリリースおよびコンフィグレーションを指定すると、指定した 2 つのコンフィグレーションのリソースが相互に比較されます。

注意：ローカルリソースを、そのリソースに割り当てられているコンフィグレーションと比較する場合は、compare コマンドを使用します。

差分が検出された各リソースについて、1 行の情報が出力されます。変更されていないリソースについては、出力は生成されません。出力行は、変更マーカーで始まります。変更マーカーの値は、以下のとおりです。

U – コンフィグレーション 1 でリソースが更新されました。

u – コンフィグレーション 2 でリソースが更新されました。

D – コンフィグレーション 1 でリソースが削除されました。

d – コンフィグレーション 2 でリソースが削除されました。

A – コンフィグレーション 1 でリソースが追加されました。

a – コンフィグレーション 2 でリソースが追加されました。

「コンフィグレーション 2」は、指定した 2 つめのコンフィグレーション、またはローカルワークスペースに割り当てられている作業コンフィグレーションです。

これらのマーカーは、組み合わせて表示されることもあります。

変更マーカーの後に、リソースの種類（ファイルまたはフォルダー）が表示されます。リソースがファイルの場合、その後にベースバージョンが表示されます。ベースバージョンは通常、比較対象バージョンの最新の共通先行バージョンになります。

コマンド構文：

```
escm rdiff [-r] <リリース名> <コンフィグレーション名> [<リリース名 2>
      <コンフィグレーション名 2>]
```

-r
: コンフィグレーションのルートおよびルートに含まれるすべてのフォルダーについて、比較を実行します。このオプションを指定しない場合、コンフィグレーションのルートフォルダーだけが比較されます。

<リリース名>
: 比較するコンフィグレーションが含まれているリリースの名前。

<コンフィグレーション名>
: 比較するコンフィグレーションの名前。

<リリース名 2>
: 最初のコンフィグレーションと比較するコンフィグレーションが含まれているリリースの名前。

<コンフィグレーション名 2>
: 最初のコンフィグレーションと比較するコンフィグレーションの名前。

例

リリース「2.0」を、ローカルワークスペースに割り当てられているリリース「1.0」と比較します。

```
C:¥test¥projects¥Project B>escm rdiff 2.0 MAIN
```

2 つのコンフィグレーションを相互に比較します。

```
C:¥test¥projects¥Project B>escm rdiff 1.0 MAIN 2.0 MAIN
```

2.6.8 setconfst – コンフィグレーションステートの設定 [SCM]

SCM コンフィグレーションのステートを設定します。

ソフトウェア開発プロセスはいずれも、組織固有のライフサイクルに基づいています。つまり、各コンフィグレーションは、ある時点での 1 つの定義されたステートになります。

作業コンフィグレーションは、通常、「in development（開発中）」のステートですが、保護されたコンフィグレーションは、「quality assurance（品質保証）」または「pre-production test（試作テスト）」のステートになります。また、日常のビルドなどの未使用コンフィグレーションには最終ステートの「archived（アーカイブ済み）」が割り当てられ、リリース済みコンフィグレーションは「released（リリース済み）」または「delivered（提供済み）」のステートになります。

ステートは、ソフトウェア開発プロセスの理解および管理に役立つとともに、時間経過によるリリースの進展を把握できます。

SCM は、コンフィグレーションステートの定義、操作、視覚化をサポートしています。ステートおよびステートの移行は、既存の開発プロセスに最適かつ柔軟に統合できるように、カスタマイズすることができます。

コマンド構文：

```
escm {setconfst | scs} -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -r <リリース名> -c <コンフィグレーション名>
      -z <ステート名> [-po <ポート>]
```

`-s <サーバ名>`
リポジトリが存在するサーバの名前。

`-d <リポジトリ名>`
プロジェクトが含まれているリポジトリの名前。

`-p <プロジェクト名>`
コンフィグレーションを格納しているプロジェクトの名前。

`-r <リリース名>`
コンフィグレーションを格納しているリリースの名前。

`-c <コンフィグレーション名>`
ステートを変更するコンフィグレーションの名前。

`-z <ステート名>`
コンフィグレーションに設定するステートの名前。

`-po <ポート>`
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、`9500` です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

例

スナップショット「Build 1_pre」に、ステート「qa」を設定します。

```
C:¥>escm setconfst -s estescmtstxp -d Sample -p "Project B" -r 1.0
-c "Build 1_pre" -z qa
```

2.6.9 updconf – コンフィグレーションプロパティの更新 [SCM、DMS]

SCM コンフィグレーション、DMS ライブラリ、またはライブラリエディションのプロパティまたはフラグを変更します。

コマンド構文：

```
escm updconf -s <サーバ名> -d <リポジトリ名> -p <プロジェクト名>
      -r <リリース名> -c <コンフィグレーション名> [-po <ポート>
      -m <コメント> -prp <プロパティ;...>]
```

`-s <サーバ名>`
リポジトリが存在するサーバの名前。

`-d <リポジトリ名>`
プロジェクトまたはライブラリが含まれているリポジトリの名前。

`-p <プロジェクト名>`
プロジェクトまたはライブラリの名前。

`-r <リリース名>`
コンフィグレーションを格納しているリリースの名前。

`-c <コンフィグレーション名>`
更新するコンフィグレーションまたはライブラリエディションの名前。

`-po <ポート>`
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、`9500` です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

`-m <コメント>`
コンフィグレーションの短い説明（スナップショットのみ）。

`-prp <プロパティ>`

このオプションは今版では利用できません。

例

コンフィグレーションにコメントを設定します。

```
C:¥>escm updconf -s estescmtstxp -d Sample -p "Project B"
-r 1.0 -c MAIN -m “Sample Comment”
```

2.6.10 setconfiglock – コンフィグレーションのロック [SCM]

コンフィグレーションをロックするか、またはロックを解除します。

SCM では、コンフィグレーション全体を迅速にロックして、コンフィグレーション内のすべてのリソースを排他的に使用できます。コンフィグレーションをロックしたユーザーだけが、そのコンフィグレーション上で、変更のコミット、スナップショットの作成、または他の書き込み処理を実行できます。この機能は、他のユーザーにコンフィグレーションの内容を変更させずに、ビルド作成、ビルド結果のコミット、および最終スナップショットの作成を確実に実行する必要があるビルドプロセスに特に役立ちます。

コンフィグレーションのロックは、別のユーザーがコンフィグレーション内のリソースをすでにロックしていても設定できます。コンフィグレーションのロックを解除すると、以前に設定されていたリソースのロックが再び有効になります。

コマンド構文：

```
escm {setconfiglock | scl} [-del] -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -r <リリース名> -c <コンフィグレーション名>
      [-po <ポート>]
```

-del
　コンフィグレーションのロックを解除します。

-s <サーバ名>
　リポジトリが存在するサーバの名前。

-d <リポジトリ名>
　プロジェクトが含まれているリポジトリの名前。

-p <プロジェクト名>
　コンフィグレーションを格納しているプロジェクトの名前。

-r <リリース名>
　コンフィグレーションを格納しているリリースの名前。

-c <コンフィグレーション名>
　ロックを設定する、またはロックを解除するコンフィグレーションの名前。

-po <ポート>
　リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

例

コンフィグレーションを排他的に使用するためにロックします。

```
C:¥>escm setconfiglock -s estescmtstxp -d Sample -p "Project B" -r 1.0
-c MAIN
```

2.6.11 monlist – 登録されたリソース監視の一覧表示 [SCM]

指定したコンフィグレーションに定義されているすべてのローカルリソース監視を一覧表示します。

コマンド構文：

```
escm monlist -s <サーバ名> -d <リポジトリ名> -p <プロジェクト名>
      -r <リリース名> -c <コンフィグレーション名> [-po <ポート>]
```

-s <サーバ名>
　リポジトリが存在するサーバの名前。

-d <リポジトリ名>
　プロジェクトが含まれているリポジトリの名前。

-p <プロジェクト名>
　コンフィグレーションを格納しているプロジェクトの名前。

-r <リリース名>
　コンフィグレーションを格納しているリリースの名前。

-c <コンフィグレーション名>
　リソース監視を一覧表示するコンフィグレーションの名前。

-po <ポート>
　リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、

9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

例

指定したコンフィグレーションのすべてのローカルリソース監視を一覧表示します。

```
C:¥>escm monlist -s esttstxp -d Sample2 -p "Project A" -r 3.0 -c MAIN
```

2.7 SCMのリリース管理

SCM プロジェクトは、リリースで構成されます。プロジェクトを SCM 内で共有すると、リリース「1.0」、作業コンフィグレーションの「MAIN」、およびベースコンフィグレーションの「BASE」が自動的に作成されます。プロジェクトの最初のリリースでは、ベースコンフィグレーションは空となっています。その他のリリースでは、新しいリリースの作成元コンフィグレーションがベースコンフィグレーションになります。リリースには、各種のビルドまたはブランチとなるような、多数のコンフィグレーションを設定できます。リリース名とコンフィグレーション名は、変更できます。

リリースは、スナップショットに基づいて作成できます。リリースは、コンフィグレーションをグループ化した論理的なオブジェクトです。SCM では、以下の 3 種類のリリースを区別しています。

- **フォローアップリリース**

  フォローアップリリースは、開発スレッドでの直系リリースです。1 つ前のリリースのスナップショットに基づいて作成されます。

- **ブランチリリース**

  ブランチリリースは、リリース内のブランチです。同じスナップショットに基づいて、異なるブランチを作成できます。これにより、並行開発が可能になります。例えば、ブランチリリースを使用して 1 つのリリースのサービスパックと修正を管理しながら、新しい（フォローアップ）リリースの開発を同時進行できます。

- **リンク済みリリース**

  リンク済みリリースは、別の SCM プロジェクト、または SCM の管理対象外プロジェクト用のリリースです。リリースどうしをリンクすることによって、プロジェクト間の依存関係をモデル化して SCM に保存し、マニュアル制作などに利用できます。

  リンク済みリリースは、「閲覧可能」にすることができます。閲覧可能なリリースはリポジトリ内に存在し、他のリリースと同様にたどることができます。「閲覧可能」ではないリンク済みリリースは、通常、リポジトリには存在しません。プロジェクトまたはプロダクトの名前とリリース名が、SCM プロジェクトに定義されるだけです。SCM リリースをリンクして、「閲覧可能」にしないこともできます。

LCM では、アプリケーションの各種バージョンとリリースを作成、計画、および監視するために、リリースを定義することもできます。LCM のリリースの内容は通常、SCM のコンフィグレーションへのリンクを作成することによって定義されます。LCM と SCM のリンクは、SCM のコンフィグレーション名が変更されたり、SCM リリースが分割されたりした場合でも、常に最新状態を保持します。

2.7.1 crearel – リリースの作成 [SCM]

指定したコンフィグレーションに基づいて、SCM プロジェクトのリリースを作成します。変更管理が有効で、SCM リポジトリ用に設定されている場合、LCM に対応するリリースが作成されます。

リリース名の制限

各種のクライアントオペレーティングシステムを適正にサポートするために、リリースの名前には以下の文字と同様、すべての制御文字を使用できません。
`* : ¥ / ? < > | "`
リリース名にこれらの文字が含まれていると、リリースの作成に失敗します。

コマンド構文：

```
escm crearel [-b –l -nolcm] -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -r <リリース名> -c <コンフィグレーション名>
      [-po <ポート> -m <コメント>] <新しいリリース名>
```

-b

新しいリリースをブランチリリースとして作成します。このオプションを指定しないと、フォローアップリリースが作成されます。

フォローアップリリース：
新しいリリースは、作成元になるベースリリースの後継リリースになります。フォローアップリリースは、開発スレッドにおいて、前のリリースの上に表示されます。ベースリリースにフォローアップリリースがすでに定義されている場合、このオプションは使用できません。スレッド内の各リリースには、最新のリリースを除き、1 つのフォローアップリリースがあります。

ブランチリリース：
このオプションを指定すると、新しいリリースが、作成元のベースリリースの下に位置付けられます。新しいリリースに基づいて、新しいブランチのフォローアップリリースを定義できます。

-l

既存のリリースを 2 つのリリースに分割します。LCM に対応するリリースが作成され、それに応じてコンフィグレーションのリンクが更新されます。

このオプションは、リリース内に 2 つ以上のスナップショットがある場合に使用できます。「-c」オプションで指定したコンフィグレーション（スナップショット）が、分割ポイントとして使用されます。コンフィグレーションおよびそのすべての後継コンフィグレーションが新しいリリースに移行し、フォローアップリリースとして作成されます。先行コンフィグレーション（スナップショット）は、古いリリース内に存続します。

リリースの分割は、既存のプロジェクトをリリース 1.0 から 2.0 に移行したい場合などに役立ちます。リリース 1.0 には、すべてのアーカイブ済みスナップショットと現在の作業コンフィグレーションが、新しい順に保存されます。任意のスナップショットを使用して、リリース 1.0 をリリース 1.0 と 2.0 に分割できます。選択したスナップショットと、そのスナップショットより新しいすべてのスナップショット、および作業コンフィグレーションが、新しいリリース 2.0 に移行します。リリース 1.0 には、それより前のスナップショットが保持されます。

リリースを分割するには、以下の条件が満たされている必要があります。

- 指定するコンフィグレーションがスナップショットであること。
- このスナップショットを含むリリースに、後継リリースが定義されていないこと。
- 指定したコンフィグレーションが、そのリリースの最初のコンフィグレーションではないこと。

-nolcm

LCM への接続を無効にします。この場合、LCM の対応するリリースは作成されません。「変更管理への自動反映」がリポジトリで有効になっている場合は（chmansett コマンドを参照）、このオプションで接続を無効にできません。

-s <サーバ名>

リポジトリが存在するサーバの名前。

-d <リポジトリ名>
プロジェクトが含まれているリポジトリの名前。

-p <プロジェクト名>
リリースが含まれているプロジェクトの名前。

-r <リリース名>
ベースコンフィグレーションが含まれているリリースの名前。これが、新しいリリースの先行リリースになります。

-c <コンフィグレーション名>
新しいリリースのベースになるコンフィグレーションの名前。指定したコンフィグレーションが、新しいリリースのベースコンフィグレーション（「BASE」）になります。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-m <コメント>
新しいリリースの短い説明。

<新しいリリース名>
新しいリリースの名前。

例

リリース「2.0」のコンフィグレーション「build 2.1」に基づいて、新しいフォローアップリリース「3.0」を作成します。

```
C:¥>escm crearel -s estescmtstxp -d Sample -p "Project B" -r 2.0 -c
"Build 2.1" -m "Enterprise Version" 3.0
```

2.7.2 closerel – リリースのクローズ [SCM]

SCM プロジェクトの指定リリースをクローズして、指定したコンフィグレーションを「released（リリース済み）」としてマークします。変更管理への自動反映が設定されている場合、LCM 側の対象となるリリースのプロセスを完了させることができます。

クローズしたリリースは、いかなる変更もコミットできません。

コマンド構文：

```
escm closerel [-nolcm] -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -r <リリース名> [-po <ポート>
      -c <コンフィグレーション名>]
```

-nolcm
LCM への接続を無効にします。この場合、LCM 側の対象となるリリースのプロセスは完了しません。「変更管理への自動反映」がリポジトリで有効になっている場合は（chmansett コマンドを参照）、このオプションで接続を無効にできません。

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
プロジェクトが含まれているリポジトリの名前。

-p <プロジェクト名>
リリースが含まれているプロジェクトの名前。

-r <リリース名>
クローズするリリースの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-c <コンフィグレーション名>
「released」としてマークするコンフィグレーションの名前。このコンフィグレーションには、このリリースの最終リソースバージョンが含まれている必要があります。コンフィグレーションを指定しない場合、リリースの作業コンフィグレーションが使用されます。複数の作業コンフィグレーションが存在すると、コマンドは中断され、対応するエラーメッセージが表示されます。

例

現在の「MAIN」コンフィグレーションをリリース済みコンフィグレーションとし、リリース「2.0」をクローズします。

```
C:¥>escm closerel -s estescmtstxp -d Sample -p "Project B" -r 2.0
-c MAIN
```

2.7.3 delrel – リリースの削除 [SCM]

SCM プロジェクトから指定したリリースを削除します。「-keepcfgs」オプションを指定しないと、リリースのすべてのコンフィグレーションが削除されます。リソースバージョンは削除されません。変更管理への自動反映が設定されている場合、LCM 側の対象となるリリースも削除されます。

コマンド構文：

```
escm delrel [-f -keepcfgs -nolcm] -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -r <リリース名> [-po <ポート>]
```

-f
確認を要求せずに、リリースを削除します。

-keepcfgs
すべてのコンフィグレーションを保持し、先行リリースに割り当てます。このオプションにより、「リリース分割」の操作を元に戻すことができます（「crearel コマンド」を参照）。

-nolcm
LCM への接続を無効にします。この場合、LCM 側の対象となるリリースは削除されません。「変更管理への自動反映」がリポジトリで有効になっている場合は（chmansett コマンドを参照）、このオプションで接続を無効にできません。

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
プロジェクトが含まれているリポジトリの名前。

-p <プロジェクト名>
リリースが含まれているプロジェクトの名前。

-r <リリース名>
削除するリリースの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

例

リリース「3.0」を削除します。

```
C:¥>escm delrel -s estescmtstxp -d Sample -p "Project B" -r 3.0
```

2.7.4 reactrel – リリースの再活性化 [SCM]

クローズしたリリースを再活性化します。クローズしたリリースに追加の作業が必要な場合、そのリリースを再活性化できます。

再活性化したリリースには、作業コンフィグレーションが存在しません。作業コンフィグレーションを作成するには、新しいブランチを作成する必要があります。

コマンド構文：

```
escm reactrel [-nolcm] -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -r <リリース名> [-po <ポート>]
```

-nolcm
LCM への接続を無効にします。この場合、LCM 側の対象となるリリースは再開しません。「変更管理への自動反映」がリポジトリで有効になっている場合は（chmansett コマンドを参照）、このオプションで接続を無効にできません。

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
プロジェクトが含まれているリポジトリの名前。

-p <プロジェクト名>
リリースが含まれているプロジェクトの名前。

-r <リリース名>
再活性化するリリースの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

例

クローズしたリリース「2.0」を再活性化します。

```
C:¥>escm reactrel -s estescmtstxp -d Sample -p "Project B" -r 2.0
```

2.7.5 renrel – リリースの名前変更 [SCM]

リリースの名前を変更します。変更管理が有効で、SCM リポジトリ用に設定されている場合、LCM の関連リリースの名前も変更されることがあります。

リリース名の制限

各種のクライアントオペレーティングシステムを適正にサポートするために、リリースの名前には以下の文字と同様、すべての制御文字を使用できません。
* : ¥ / ? < > | "
リリース名にこれらの文字が含まれていると、リリースの名前変更に失敗します。

コマンド構文：

```
escm renrel [-nolcm] -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -r <リリース名> [-po <ポート>]
      <新しいリリース名>
```

-nolcm
LCM への接続を無効にします。この場合、LCM の関連するリリースの名前は変更されません。「変更管理への自動反映」がリポジトリで有効になっている場合は（chmansett コマンドを参照）、このオプションで接続を無効にできません。

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
プロジェクトが含まれているリポジトリの名前。

-p <プロジェクト名>
リリースが含まれているプロジェクトの名前。

-r <リリース名>
名前を変更するリリースの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

<新しいリリース名>
変更後のリリースの名前。

例

リリース「2.0」の名前を「2.1」に変更します。

```
C:¥>escm renrel -s estescmtstxp -d Sample -p "Project B" -r 2.0 2.1
```

2.7.6 movrel – リリースの移動 [SCM]

1 つのリリースと、その後継のすべてのフォローアップリリースとブランチリリースを、リリース構造の別の場所に移動します。

コマンド構文：

```
escm movrel -s <サーバ名> -d <リポジトリ名> -p <プロジェクト名>
      -r <リリース名> -r2 <リリース名 2> -b <タイプ> [-po <ポート>]
```

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
プロジェクトが含まれているリポジトリの名前。

-p <プロジェクト名>
リリースが含まれているプロジェクトの名前。

-r <リリース名>
移動するリリースの名前。

-r2 <リリース名 2>
移動するリリースの新しい先行リリースとなるリリースの名前。

-b <タイプ>
移動後のリリースのタイプ。ブランチリリースとして定義するには値に「1」を指定し、フォローアップリリースとして定義するには「0」を指定します（リリースタイプの詳細は、「crearel コマンド」を参照）。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

例

リリース「2.3」を移動し、リリース「2.2」のフォローアップリリースとして設定します。

```
C:¥>escm movrel -s estescmtstxp -d Sample -p "Project B" -r 2.3
–r2 2.2 –b 0
```

2.7.7 setreldescr – リリースの説明とタイプの設定 [SCM]

リリースの短い説明（コメント）を更新します。このコマンドを使用して、リリースのタイプを指定することもできます。

コマンド構文：

```
escm setreldescr -s <サーバ名> -d <リポジトリ名> -p <プロジェクト名>
      -r <リリース名> [-po <ポート> -m <コメント> -b <タイプ>
      -prp <プロパティ;...>]
```

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
プロジェクトが含まれているリポジトリの名前。

-p <プロジェクト名>
リリースが含まれているプロジェクトの名前。

-r <リリース名>
コメントを更新するリリースの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-m <コメント>
リリースの新しい説明。

-b <タイプ>
ブランチリリースとして定義するには値に「1」を指定し、フォローアップリリースとして定義するには「0」を指定します（リリースタイプの詳細は、「crearel コマンド」を参照）。

-prp <プロパティ>

このオプションは今版では利用できません。

例

リリース「2.1」に新しい説明を設定します。

```
C:¥>escm setreldescr -s estescmtstxp -d Sample -p "Project B" -r 2.1
-m "Emergency release"
```

リリース「3.0」をフォローアップリリースとして定義します。

```
C:¥>escm setreldescr -s estescmtstxp -d Sample -p "Project B" -r 3.0
-b 0
```

2.7.8 asssubrel – リリースの割当て [SCM]

SCM プロジェクトまたはプロダクトのリリースに、サブリリースを割り当てます。サブリリースは、プロジェクトやプロダクトのリンク済みリリース、またはプロダクトのメンバーリリースのいずれかです。

サブリリースは、別の SCM プロジェクトやプロダクトのリリース、または SCM の管理対象外のプロジェクトやプロダクトになります。サブリリースが SCM プロジェクトやプロダクトのリリースである場合、「閲覧可能」として定義し、他のリリースと同様にたどることができます。閲覧可能なサブリリースの場合、「-e」オプション、「-f」オプション、「-c」オプションにオリジナルのリリースおよびコンフィグレーションを指定し、「-a」オプションと「-b」オプションを指定する必要があります。

「–a」オプションと「–b」オプションを指定しないと、サブリリースは「閲覧不可」として定義されます。この場合、サブリリースが SCM の管理対象かどうかに関係なく、「–e」オプションと「–f」オプションに任意の文字列を指定できます。

コマンド構文：

```
escm {asssubrel | assr} -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -r <リリース名> -e <サブプロジェクト名>
      -f <サブリリース名> [-po <ポート> -a <サブサーバ名>
      -pod <サブポート> -b <サブリポジトリ名>
      -c <サブコンフィグレーション名> -t <ターゲット名>]
```

-s <サーバ名>
: リポジトリが存在するサーバの名前。

-d <リポジトリ名>
: プロジェクトが含まれているリポジトリの名前。

-p <プロジェクト名>
: サブリリースを割り当てるリリースが含まれているプロジェクトまたはアプリケーションの名前。

-r <リリース名>
: サブリリースを割り当てるリリースの名前。

-e <サブプロジェクト名>
: サブリリースとして割り当てるリリースが含まれているプロジェクトまたはアプリケーションの名前。サブリリースを閲覧可能にする場合、SCM プロジェクトまたはプロダクトを指定する必要があります。サブリリースを閲覧不可にする場合は、任意の文字列を指定できます。

-f <サブリリース名>
: サブリリースとして割り当てるリリースの名前。サブリリースを閲覧可能にする場合、SCM プロジェクトまたはプロダクトのリリースで、「–c」オプションで指定し

たコンフィグレーションに含まれている必要があります。サブリリースを閲覧不可にする場合は、任意の文字列を指定できます。

`-po <ポート>`
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

`-a <サブサーバ名>`
サブリリースが含まれているリポジトリが存在するサーバの名前。サブリリースを閲覧可能にする場合、このオプションを指定する必要があります。

`-pod <サブポート>`
サブリリースが含まれているリポジトリが存在するサーバにアクセスするポート。指定しない場合、「-po」オプションで指定したポートが使用されます。

`-b <サブリポジトリ名>`
サブリリースが含まれているリポジトリの名前。サブリリースを閲覧可能にする場合、このオプションを指定する必要があります。

`-c <サブコンフィグレーション名>`
リリースに割り当てるサブリリースのコンフィグレーションの名前。サブリリースを閲覧可能にする場合、このオプションを指定する必要があります。

`-t <ターゲット名>`
バッチ方式でサブリリースをチェックアウトする場合、またはサブリリースを割り当てたアプリケーションをエクスポートする場合に、サブリリース用に使用するルートフォルダーの名前。デフォルトでは、サブプロジェクト（「-e」オプション）の名前が使用されます。

例

プロジェクト「Demo」のリリース「5.00」に、閲覧不可のサブリリース「Java JDK 1.4.2」を定義します。

```
C:¥>escm asssubrel -s estenabler1 -d SCM1 -p Demo -r 5.00 -e "Java JDK"
   -f 1.4.2
```

2.7.9 deasssubrel – リリースの割当て解除 [SCM]

「-p」オプションと「-r」オプションで指定した SCM プロジェクトまたはプロダクトリリースから、「-e」オプションと「-f」オプションで指定したサブリリースの割当てを解除します。

コマンド構文：

```
escm {deasssubrel | dsr} -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -r <リリース名> -e <サブプロジェクト名>
      -f <サブリリース名> [-po <ポート>]
```

`-s <サーバ名>`
リポジトリが存在するサーバの名前。

`-d <リポジトリ名>`
プロジェクトまたはプロダクトが含まれているリポジトリの名前。

`-p <プロジェクト名>`
サブリリースを解除するリリースが含まれているプロジェクトまたはプロダクトの名前。

-r <リリース名>
　サブリリースを解除するリリースの名前。

-e <サブプロジェクト名>
　解除するサブリリースが含まれているプロジェクトまたはプロダクトの名前。

-f <サブリリース名>
　リリースから解除するサブリリースの名前。

-po <ポート>
　リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

例

プロジェクト「Project B」のリリース「2.1」から、閲覧可能なサブリリース「Project C 1.0」の割当てを解除します。

```
C:¥>escm deasssubrel -s estescmtstxp -d Sample -p "Project B" -r 2.1 -e
"Project C" -f 1.0
```

2.7.10 listsubrel – 割当て済みリリースのリスト [SCM]

指定した SCM プロジェクトまたはプロダクトリリースに割り当てられている、すべてのサブリリースを一覧表示します。各サブリリースについて、閲覧可能かどうかの情報が表示されます。

コマンド構文：

```
escm {listsubrel | lsr} -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -r <リリース名> [-po <ポート>]
```

-s <サーバ名>
　リポジトリが存在するサーバの名前。

-d <リポジトリ名>
　プロジェクトまたはプロダクトが含まれているリポジトリの名前。

-p <プロジェクト名>
　サブリリースを一覧表示するリリースが含まれているプロジェクトまたはプロダクトの名前。

-r <リリース名>
　サブリリースを一覧表示するリリースの名前。

-po <ポート>
　リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

例

「Project B」のリリース「2.1」のサブリリースを表示します。

```
C:¥>escm listsubrel -s estescmtstxp -d Sample -p "Project B" -r 2.1
```

2.8 ドキュメントの管理

DMS は SCM をベースにしています。したがって、SCM で使用できるコマンドの多くは、DMS にも適用されます。以下の機能については、edms.bat バッチファイルを使用して呼び出す DMS 独自のコマンドが提供されています。

- ドキュメントのライフサイクル
- レポート機能

変更管理が有効で DMS リポジトリ用に設定されていれば、DMS でタスクの操作をサポートできます。

以下のセクションでは、これらのコマンドについて説明します。

2.8.1 createlibrary – ライブラリの作成 [DMS]

指定したリポジトリに、新しい DMS ライブラリを作成します。

変更管理が有効で、DMS リポジトリで設定されている場合、LCM に対応するアプリケーションおよびリリースが作成されます。LCM アプリケーションでは、必要に応じてコマンドを実行したユーザーに 1 つ以上のユーザーロールが割り当てられます。これは、ユーザーに割り当てられている SCM ユーザーロールに含まれるすべての関連するオーソリティを LCM オーソリティにマップすることで実施します。関連するオーソリティは、SCM アクションを LCM に反映させるために必要となるオーソリティです。

SCM または DMS と LCM 間の関連するオーソリティのマッピングの詳細については、「2.4 ロールとオーソリティ」を参照してください。

命名の制限

各種のクライアントオペレーティングシステムを適正にサポートするために、ライブラリの名前にはすべての制御文字および以下の文字を使用できません。
* : ¥ / ? < > | "
ライブラリ名にこれらの文字が含まれていると、ライブラリの作成に失敗します。

コマンド構文：

```
edms {createlibrary | crelib} [-nolcm] -s <サーバ名>
      -d <リポジトリ名> -p <プロジェクト名> -g <グループ名>
      [-po <ポート>]
```

-nolcm
: LCM への接続を無効にします。この場合、LCM のアプリケーションとリリースは作成されません。「変更管理への自動反映」がリポジトリで有効になっている場合は（chmansett コマンドを参照）、このオプションで接続を無効にできません。

-s <サーバ名>
: リポジトリが存在するサーバの名前。

-d <リポジトリ名>
: ライブラリを保存するリポジトリの名前。

-p <プロジェクト名>
: 新しいライブラリの名前。

-g <グループ名>
: リポジトリサーバが認識している最大 8 のユーザーグループを、セミコロンで区切って指定します。指定したユーザーグループのメンバーは、新しいライブラリで作業ができます。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

例

「soopl」というサーバ上のリポジトリ「Sample」に、ライブラリ「Sample A」を作成します。

```
C:¥>edms createlibrary -s soop1 -d Sample -p "Sample A" –g
"GroupA;GroupB"
```

2.8.2 createlibrarygroup – ライブラリグループの作成 [DMS]

指定したリポジトリに、新しい DMS ライブラリグループを作成します。

ライブラリグループは、SCM プロダクトのリリースにマップされます。ライブラリグループが作成されると、リリース"1.0"を持つ SCM プロダクトが作成されます。

命名の制限

各種のクライアントオペレーティングシステムを適正にサポートするために、ライブラリグループの名前にはすべての制御文字および以下の文字を使用できません。
* : ¥ / ? < > | "
ライブラリグループ名にこれらの文字が含まれていると、ライブラリグループの作成に失敗します。

コマンド構文：

```
edms {createlibrarygroup | crelibgrp} -s <サーバ名>
      -d <リポジトリ名> -p <プロジェクト名> -g <グループ名>
      [-po <ポート>]
```

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
ライブラリグループを含むリポジトリの名前。

-p <プロジェクト名>
新しいライブラリグループの名前。リポジトリに同じ名前の SCM プロダクトが存在する場合は、エラーメッセージが表示されます。このような場合、ライブラリグループに対して別の名前を指定するか、別のリポジトリにライブラリグループを作成します。

-g <グループ名>
リポジトリサーバが認識している最大 8 のユーザーグループを、セミコロンで区切って指定します。指定したユーザーグループのメンバーは、新しいライブラリグループで作業ができます。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

例

「soopl」というサーバ上のリポジトリ「Sample」に、ライブラリグループ「Sample Group A」を作成します。

```
C:¥>edms createlibrarygroup -s soop1 -d Sample -p "Sample Group A" -g
"GroupA;GroupB"
```

2.8.3 assigngroup – ライブラリグループへのアイテムの割当て [DMS]

ライブラリまたはライブラリグループに指定したターゲットとなるライブラリグループを割り当てます。

ターゲットとなるライブラリグループは「-s」、「-d」、「-p」、「-po」および「-r」オプションにより指定されます。割当て対象となるライブラリまたはライブラリグループは「-sd」、「-dd」、「-pd」、「-pod」、「-rd」および「-cd」オプションにより指定されます。

注意: 1 つのライブラリグループには、各 SCM プロジェクトに対して、関連するライブラリを 1 つだけ含めることができます。ライブラリをライブラリグループに割り当てると、同じ SCM プロジェクトに関連付けられている既存のライブラリは、ライブラリグループから自動的に削除されます。

コマンド構文：

```
edms {assigngroup | assgrp} -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> [-po <ポート> -r <リリース名> -sd <サーバ名>
      -dd <リポジトリ名>] -pd <プロジェクト名> [-pod <ポート> -rd <リ
      リース名>
      -cd <コンフィグレーション名> -eud <ユーザー名> -epd <パスワード>]
```

`-s <サーバ名>`
ターゲットとなるライブラリグループを含むサーバの名前。

`-d <リポジトリ名>`
ターゲットとなるライブラリグループを含むリポジトリの名前。.

`-p <プロジェクト名>`
ターゲットなるライブラリグループの名前。

`-po <ポート>`
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、`9500` です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

`-r <リリース名>`
ターゲットとなるライブラリグループが使用している SCM リリースの名前。指定しない場合、最初に検出されたリリースが使用されます。

`-sd <サーバ名>`
割当て対象となるライブラリまたはライブラリグループを含むリポジトリが存在するサーバの名前。指定しない場合、「-s」オプションで指定されるサーバが使用されます。

`-dd <リポジトリ名>`
割当て対象となるライブラリまたはライブラリグループを含むリポジトリの名前。指定しない場合、「-d」オプションで指定されるリポジトリが使用されます。

`-pd <プロジェクト名>`
割当て対象となるライブラリまたはライブラリグループの名前。

`-pod <ポート>`
割当て対象となるライブラリまたはライブラリグループを含むリポジトリが存在す

るサーバにアクセスするポート。指定しない場合、「-po」オプションで指定されるポートが使用されます。

`-rd <リリース名>`
割当て対象となるライブラリまたはライブラリグループが使用する SCM リリースの名前。指定しない場合、「-r」オプションで指定されるリリースが使用されます。

`-cd <コンフィグレーション名>`
割当て対象となるライブラリが使用している SCM コンフィグレーションの名前。指定しない場合、最初に検出されたリリースが使用されます。

`-eud <ユーザー名>`
割当て対象となるライブラリまたはライブラリグループを含むリポジトリへの接続に利用するユーザー名。指定しない場合、リポジトリへ接続しているユーザーが使用されます。

`-epd <パスワード>`
割当て対象となるライブラリまたはライブラリグループを含むリポジトリへの接続に利用するパスワード。

例

サーバ「soopl」のリポジトリ「Sample」に存在するライブラリ「Sample A」に、ターゲットとなるライブラリグループ「Sample Group A」を割り当てます。

```
C:¥>edms assigngroup -s soop1 -d Sample -p "Sample Group A" -pd "Sample
A"
```

2.8.4 deassigngroup – ライブラリグループからのアイテムの割当て解除 [DMS]

ライブラリまたはライブラリグループから、指定したターゲットとなるライブラリグループの割当てを解除します。

ターゲットとなるライブラリグループは「-s」、「-d」、「-p」、「-po」および「-r」オプションにより指定されます。割当てを解除するライブラリまたはライブラリグループは「-sd」、「-dd」、「-pd」、「-pod」、「-rd」および「-cd」で指定します。

コマンド構文：

```
edms {deassigngroup | dgrp} -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> [-po <ポート> -r <リリース名> -sd <サーバ名>
      -dd <リポジトリ名>] -pd <プロジェクト名> [-pod <ポート> -rd <リ
      リース名>
      -cd <コンフィグレーション名> -eud <ユーザー名> -epd <パスワード>]
```

`-s <サーバ名>`
ターゲットとなるライブラリグループを含むサーバの名前。

`-d <リポジトリ名>`
ターゲットとなるライブラリグループを含むリポジトリの名前。.

`-p <プロジェクト名>`
ターゲットなるライブラリグループの名前。

`-po <ポート>`
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、`9500` です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-r <リリース名>
ターゲットとなるライブラリグループが使用している SCM リリースの名前。指定しない場合、最初に検出されたリリースが使用されます。

-sd <サーバ名>
割当てを解除するライブラリまたはライブラリグループを含むリポジトリが存在するサーバの名前。指定しない場合、「-s」オプションで指定されるサーバが使用されます。

-dd <リポジトリ名>
割当てを解除するライブラリまたはライブラリグループを含むリポジトリの名前。指定しない場合、「-d」オプションで指定されるリポジトリが使用されます。

-pd <プロジェクト名>
割当て対象となるライブラリまたはライブラリグループの名前。

-pod <ポート>
割当てを解除するライブラリまたはライブラリグループを含むリポジトリが存在するサーバにアクセスするポート。指定しない場合、「-po」オプションで指定されるポートが使用されます。

-rd <リリース名>
割当てを解除するライブラリまたはライブラリグループが使用する SCM リリースの名前。指定しない場合、「-r」オプションで指定されるポートが使用されます。

-cd <コンフィグレーション名>
割当てを解除するライブラリが使用している SCM コンフィグレーションの名前。指定しない場合、最初に検出されたリリースが使用されます。

-eud <ユーザー名>
割当てを解除するライブラリまたはライブラリグループを含むリポジトリへの接続に利用するユーザー名。指定しない場合、リポジトリへ接続しているユーザーが使用されます。

-epd <パスワード>
割当てを解除するライブラリまたはライブラリグループを含むリポジトリへの接続に利用するパスワード。

例

サーバ「soopl」のリポジトリ「Sample」に存在するライブラリ「SampleA」から、ターゲットとなるライブラリグループ「Sample Group A」の割当てを解除します。

```
C:¥>edms deassigngroup -s soop1 -d Sample -p "Sample Group A" .pd
"Sample A"
```

2.8.5 deletelibrary – ライブラリの削除 [DMS]

指定したリポジトリから DMS ライブラリまたはライブラリグループを削除します。

ライブラリが SCM プロジェクトの作業コンフィグレーションで、SCM プロジェクトに他の作業コンフィグレーション、リリース、またはリリース内にブランチが存在しない場合は、プロジェクトが削除されます。それ以外の場合は、関連する作業コンフィグレーションだけが削除されます。

SCM プロダクトが、削除されるライブラリグループ以外にリリースを持たない場合は、SCM プロダクトは削除されます。SCM プロダクトが、他にリリースを持つ場合は、対応するリリースだけが削除されます。

削除の実行中にキャンセルすると、データの一部がリポジトリ内に残ります。この場合、再度このコマンドを実行し、ライブラリまたはライブラリグループを完全に削除してください。

注意：このアクション（削除）は元に戻すことができません。

コマンド構文：

```
edms {deletelibrary | dellib} [-f] -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> [-po <ポート> -r <リリース名>
      -c <コンフィグレーション名>]
```

`-f`
確認を要求せずに、ライブラリまたはライブラリグループを削除します。

`-s <サーバ名>`
リポジトリが存在するサーバの名前。

`-d <リポジトリ名>`
ライブラリまたはライブラリグループが含まれているリポジトリの名前。

`-p <プロジェクト名>`
削除するライブラリまたはライブラリグループの名前。

`-po <ポート>`
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、`9500` です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

`-r <リリース名>`
ライブラリが使用している SCM リリースの名前。指定しない場合、最初に検出されたリリースが使用されます。

`-c <コンフィグレーション名>`
ライブラリが使用している SCM 作業コンフィグレーションの名前。指定しない場合、リリースの最初の検出可能な作業コンフィグレーションが使用されます。

例

「soopl」というサーバ上のリポジトリ「Sample」から、ライブラリ「Sample A」を削除します。

```
C:¥>edms deletelibrary -s soop1 -d Sample -p "Sample A" –f
```

2.8.6 createedition – ライブラリエディションの作成 [DMS]

ライブラリのエディションを作成します。エディションは、ライブラリ内の現行ドキュメントバージョンの構造化された集合で、これ以上の変更はできません。

命名の制限

各種のクライアントオペレーティングシステムを適正にサポートするために、エディションの名前にはすべての制御文字および以下の文字を使用できません。
`* : ¥ / ? < > | "`
エディション名にこれらの文字が含まれていると、エディションの作成に失敗します。

コマンド構文：

```
edms {createedition | creedt} -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -m <コメント> -en <エディション名>
      [-po <ポート> -r <リリース名> -c <コンフィグレーション名>]
```

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
ライブラリが含まれているリポジトリの名前。

-p <プロジェクト名>
ライブラリの名前。

-m <コメント>
新しいエディションの短い説明。

-en <エディション名>
作成するエディションの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-r <リリース名>
ライブラリが使用している SCM リリースの名前。指定しない場合、最初の検出可能なリリースが使用されます。

-c <コンフィグレーション名>
ライブラリが使用している SCM 作業コンフィグレーションの名前。指定しない場合、リリースの最初の検出可能な作業コンフィグレーションが使用されます。

例

ライブラリ「Sample A」のエディション「My edition 1」を作成します。

```
C:¥>edms createedition -s soop1 -d Sample -p "Sample A" –m "Create an
edition" –en "My edition 1"
```

2.8.7 createdocument – ドキュメントの作成 [DMS]

指定したフォルダーまたはライブラリに、空のドキュメントを作成します。

命名の制限

各種のクライアントオペレーティングシステムを適正にサポートするために、ドキュメントの名前にはすべての制御文字および以下の文字を使用できません。
* : ¥ / ? < > | "
ドキュメント名にこれらの文字が含まれていると、ドキュメントの作成に失敗します。

コマンド構文：

```
edms {createdocument | credoc} -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -m <コメント> -dn <ドキュメント名>
      [-po <ポート> -r <リリース名> -c <コンフィグレーション名>
      -i <アイテムパス> -t <タスク ID>]
```

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
ライブラリが含まれているリポジトリの名前。

-p <プロジェクト名>
ライブラリの名前。

-m <コメント>
ドキュメント作成に関する短い説明。このコメントは、新しいドキュメントバージョンと一緒にリポジトリ内に保存されます。

-dn <ドキュメント名>
作成するドキュメントの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-r <リリース名>
ライブラリが使用している SCM リリースの名前。指定しない場合、最初の検出可能なリリースが使用されます。

-c <コンフィグレーション名>
ライブラリが使用している SCM 作業コンフィグレーションの名前。指定しない場合、リリースの最初の検出可能な作業コンフィグレーションが使用されます。

-i <アイテムパス>
新しいドキュメントを作成するフォルダーのパス名。指定しない場合、ドキュメントはライブラリのルートに作成されます。

-t <タスク ID>
新しいドキュメントを登録する 1 つ以上の LCM タスクの名前を、セミコロンで区切って指定します。ID の前部にパディングされている「0」は、省略できます。

例

フォルダー「FolderA」に、「My Document」という新しいドキュメントを作成します。

```
C:¥>edms createdocument -s soop1 -d Sample -p "Sample A" -i "/FolderA"
–m "Create a document" –dn "My Document"
```

2.8.8 createfolder – フォルダーの作成 [DMS]

指定したフォルダーまたはライブラリに、新しいフォルダーを作成します。

命名の制限

各種のクライアントオペレーティングシステムを適正にサポートするために、フォルダーの名前にはすべての制御文字および以下の文字を使用できません。
* : ¥ / ? < > | "
フォルダー名にこれらの文字が含まれていると、フォルダーの作成に失敗します。

コマンド構文：

```
edms {createfolder | crefol} -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -m <コメント> -fn <フォルダー名> [-po <ポート>
      -r <リリース名> -c <コンフィグレーション名> -i <アイテムパス>
      -t <タスク ID>]
```

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
ライブラリが含まれているリポジトリの名前。

-p <プロジェクト名>
ライブラリの名前。

-m <コメント>
フォルダー作成に関する短い説明。

-fn <フォルダー名>
作成するフォルダーの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-r <リリース名>
ライブラリが使用している SCM リリースの名前。指定しない場合、最初の検出可能なリリースが使用されます。

-c <コンフィグレーション名>
ライブラリが使用している SCM 作業コンフィグレーションの名前。指定しない場合、リリースの最初の検出可能な作業コンフィグレーションが使用されます。

-i <アイテムパス>
新しいフォルダーを作成するフォルダーのパス名。指定しない場合、フォルダーはライブラリのルートに作成されます。

-t <タスク ID>
新しいフォルダーを登録する 1 つ以上の LCM タスクの名前を、セミコロンで区切って指定します。ID の前部にパディングされている「0」は、省略できます。

例

フォルダー「FolderA」に、「My Folder」という新しいフォルダーを作成します。

```
C:¥>edms createdocument -s soop1 -d Sample -p "Sample A" -i "/FolderA"
–m "Create a folder" –fn "My Folder"
```

2.8.9 move – アイテムの移動 [DMS]

フォルダー、ドキュメント、またはショートカットを、同じまたは異なる DMS ライブラリの別の場所に移動します。

コピーするアイテムは、「-s」オプション、「-d」オプション、「-p」オプション、「–i」オプションで指定します。格納先は、「-sd」オプション、「-dd」オプション、「-pd」オプション、「-id」オプションで任意に指定します。デフォルトでは、アイテムは同じライブラリ内で移動されます。「–id」オプションを指定しない場合、アイテムはライブラリのルートに移動します。

移動の対象になるのは、現在チェックアウトされているドキュメントまたはフォルダーです。必要に応じて、ローカルワークスペースを更新できます（「–w」オプション）。

コマンド構文：

```
edms {move | mv} [-f] -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -i <アイテムパス> -m <コメント>
      [-w <ローカルワークスペース名> -po <ポート> -r <リリース名>
      -c <コンフィグレーション名> -sd <サーバ名> -dd <リポジトリ名>
      -pd <プロジェクト名> -pod <ポート> -rd <リリース名>
      -cd <コンフィグレーション名> -id <アイテムパス> -eud <ユーザー名>
      -epd <パスワード> -t <タスク ID>
```

-f
移動する同名のアイテムにより、確認なしで、格納先にある既存のアイテムを上書きします。

-s <サーバ名>
移動するアイテムが含まれているリポジトリが存在するサーバの名前。

-d <リポジトリ名>
移動するアイテムのあるライブラリが含まれているリポジトリの名前。

-p <プロジェクト名>
移動するアイテムが含まれているライブラリの名前。

-i <アイテムパス>
移動するアイテムのパス名。

-m <コメント>
移動に関する短い説明。

-w <ローカルワークスペース名>
移動するアイテムがチェックアウトされたローカルワークスペースのパス名。

-po <ポート>
送信元リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-r <リリース名>
移動するアイテムを含むライブラリが使用している SCM リリースの名前。指定しない場合、最初の検出可能なリリースが使用されます。

-c <コンフィグレーション名>
移動するアイテムを含むライブラリが使用している SCM 作業コンフィグレーションの名前。指定しない場合、リリースの最初の検出可能な作業コンフィグレーションが使用されます。

-sd <サーバ名>
アイテムの移動先リポジトリが存在する格納先サーバの名前。指定しない場合、「-s」オプションで指定したサーバが使用されます。

-dd <リポジトリ名>
アイテムの移動先ライブラリが含まれている格納先リポジトリの名前。指定しない場合、「-d」オプションで指定したリポジトリが使用されます。

-pd <プロジェクト名>
アイテムを移動する格納先ライブラリの名前。指定しない場合、「-p」オプションで指定したライブラリが使用されます。

-pod <ポート>
格納先リポジトリサーバにアクセスするポート。指定しない場合、「-po」オプションで指定したポートが使用されます。

-rd <リリース名>
アイテムの移動先ライブラリが使用している格納先 SCM リリースの名前。指定しない場合、「-r」オプションで指定したリリースが使用されます。

-cd <コンフィグレーション名>
アイテムの移動先ライブラリが使用している格納先 SCM 作業コンフィグレーションの名前。指定しない場合、「-c」オプションで指定したコンフィグレーションが使用されます。

-id <アイテムパス>
　アイテムを移動する格納先ライブラリのフォルダーのパス名。指定しない場合、アイテムは格納先ライブラリのルートに移動します。

-eud <ユーザー名>
　格納先リポジトリへの接続に使用するユーザーの名前。指定しない場合、送信元リポジトリに接続しているユーザーが使用されます。

-epd <パスワード>
　格納先リポジトリに接続するユーザーのパスワード。

-t <タスク ID>
　送信元ライブラリに移動操作を登録する 1 つ以上の LCM タスクの ID を、セミコロンで区切って指定します。タスクを指定した場合は、アイテムを他のライブラリに移動することはできません。ID の前部にパディングされている「0」は、省略できます。

例

ドキュメント「My Document」を、ライブラリ「Sample A」からライブラリ「Sample B」に移動します。

```
C:¥>edms move -s soop1 -d Sample -p "Sample A" -i "/MyDocument" –pd
"Sample B" –id "/MyDestinationFolder" –m "Move my document"
```

2.8.10 copy – アイテムのコピー [DMS]

フォルダー、ドキュメント、またはショートカットを、同じまたは異なる DMS ライブラリの別の場所にコピーします。

移動するアイテムは、「-s」オプション、「-d」オプション、「-p」オプション、「–i」オプションで指定します。格納先は、「-sd」オプション、「-dd」オプション、「-pd」オプション、「-id」オプションで任意に指定します。デフォルトでは、アイテムは同じライブラリ内でコピーされます。「–id」オプションを指定しない場合、アイテムはライブラリのルートにコピーされます。

コピーの対象になるのは、現在チェックアウトされているドキュメントまたはフォルダーです。必要に応じて、ローカルワークスペースを更新できます（「–w」オプション）。

ドキュメントまたはフォルダーを同じ場所にコピーすると、コピーは「Copy of <アイテム>」という名前になります。「<アイテム>」は、コピーしたドキュメントまたはフォルダーの元の名前です。

コマンド構文：

```
edms {copy | cp} [-f] -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -i <アイテムパス> -m <コメント>
      [-w <ローカルワークスペース名> -po <ポート> -r <リリース名>
      -c <コンフィグレーション名> -sd <サーバ名> -dd <リポジトリ名>
      -pd <プロジェクト名> -pod <ポート> -rd <リリース名>
      -cd <コンフィグレーション名> -id <アイテムパス> -eud <ユーザー名>
      -epd <パスワード> -t <タスク ID>]
```

-f
　コピーする同名のアイテムにより、確認なしで、格納先にある既存のアイテムを上書きします。

-s <サーバ名>
　コピーするアイテムが含まれているリポジトリが存在するサーバの名前。

-d <リポジトリ名>
コピーするアイテムのあるライブラリが含まれているリポジトリの名前。

-p <プロジェクト名>
コピーするアイテムが含まれているライブラリの名前。

-i <アイテムパス>
コピーするアイテムのパス名。

-m <コメント>
コピーに関する短い説明。

-w <ローカルワークスペース名>
コピーするアイテムがチェックアウトされたローカルワークスペースのパス名。

-po <ポート>
送信元リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-r <リリース名>
コピーするアイテムを含むライブラリが使用している SCM リリースの名前。指定しない場合、最初の検出可能なリリースが使用されます。

-c <コンフィグレーション名>
コピーするアイテムを含むライブラリが使用している SCM 作業コンフィグレーションの名前。指定しない場合、リリースの最初の検出可能な作業コンフィグレーションが使用されます。

-sd <サーバ名>
アイテムのコピー先リポジトリが存在する格納先サーバの名前。指定しない場合、「-s」オプションで指定したサーバが使用されます。

-dd <リポジトリ名>
アイテムのコピー先ライブラリが含まれている格納先リポジトリの名前。指定しない場合、「-d」オプションで指定したリポジトリが使用されます。

-pd <プロジェクト名>
アイテムをコピーする格納先ライブラリの名前。指定しない場合、「-p」オプションで指定したライブラリが使用されます。

-pod <ポート>
格納先リポジトリサーバにアクセスするポート。指定しない場合、「-po」オプションで指定したポートが使用されます。

-rd <リリース名>
アイテムのコピー先ライブラリが使用している格納先 SCM リリースの名前。指定しない場合、「-r」オプションで指定したリリースが使用されます。

-cd <コンフィグレーション名>
アイテムのコピー先ライブラリが使用している格納先 SCM 作業コンフィグレーションの名前。指定しない場合、「-c」オプションで指定したコンフィグレーションが使用されます。

-id <アイテムパス>
アイテムをコピーする格納先ライブラリのフォルダーのパス名。指定しない場合、アイテムは格納先ライブラリのルートにコピーされます。

-eud <ユーザー名>
格納先リポジトリへの接続に使用するユーザーの名前。指定しない場合、送信元リポジトリに接続しているユーザーが使用されます。

-epd <パスワード>
格納先リポジトリに接続するユーザーのパスワード。

-t <タスク ID>
格納先ライブラリに新しいアイテムを登録する 1 つ以上の LCM タスクの ID を、セミコロンで区切って指定します。ID の前部にパディングされている「0」は、省略できます。

例

ドキュメント「My Document」を、ライブラリ「Sample A」のフォルダー「DestFolder」にコピーします。

```
C:¥>edms copy -s soop1 -d Sample -p "Sample A" -i "/MyDocument" –w
"/workspace" –id "/DestFolder" –m "Copy my document"
```

2.8.11 createshortcut – ショートカットの作成 [DMS]

ドキュメントまたはフォルダーへのショートカットを作成します。

ショートカットで参照するアイテムは、「-s」オプション、「-d」オプション、「-p」オプション、「-i」オプションで指定します。ショートカットの場所は、「-sd」オプション、「-dd」オプション、「-pd」オプション、「-id」オプションで指定します。デフォルトでは、参照するアイテムと同じライブラリ内にショートカットが作成されます。「-id」オプションを指定しない場合、ショートカットはライブラリのルートに作成されます。

コマンド構文：

```
edms {createshortcut | cresh} -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -i <アイテムパス> -m <コメント> [-po <ポート>
      -r <リリース名> -c <コンフィグレーション名> -sd <サーバ名>
      -dd <リポジトリ名> -pd <プロジェクト名> -pod <ポート>
      -rd <リリース名> -cd <コンフィグレーション名> -id <アイテムパス>
      -eud <ユーザー名> -epd <パスワード> -t <タスク ID>]
```

-s <サーバ名>
新しいショートカットで参照するアイテムが含まれているリポジトリが存在するサーバの名前。

-d <リポジトリ名>
新しいショートカットで参照するアイテムのあるライブラリが含まれているリポジトリの名前。

-p <プロジェクト名>
新しいショートカットで参照するアイテムが含まれているライブラリの名前。

-i <アイテムパス>
新しいショートカットで参照するアイテムのパス名。

-m <コメント>
ショートカットの作成に関する短い説明。

-po <ポート>
新しいショートカットで参照するアイテムが含まれているリポジトリのサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-r <リリース名>
新しいショートカットで参照するアイテムを含むライブラリが使用している SCM リリースの名前。指定しない場合、最初の検出可能なリリースが使用されます。

-c <コンフィグレーション名>
新しいショートカットで参照するアイテムを含むライブラリが使用している SCM 作業コンフィグレーションの名前。指定しない場合、リリースの最初の検出可能な作業コンフィグレーションが使用されます。

-sd <サーバ名>
ショートカットを作成するリポジトリが存在する格納先サーバの名前。指定しない場合、「-s」オプションで指定したサーバが使用されます。

-dd <リポジトリ名>
ショートカットを作成するライブラリが含まれている格納先リポジトリの名前。指定しない場合、「-d」オプションで指定したリポジトリが使用されます。

-pd <プロジェクト名>
ショートカットを作成する格納先ライブラリの名前。指定しない場合、「-p」オプションで指定したライブラリが使用されます。

-pod <ポート>
ショートカットの作成先リポジトリがあるサーバにアクセスするポート。指定しない場合、「-po」オプションで指定したポートが使用されます。

-rd <リリース名>
ショートカットを作成するライブラリが使用している格納先 SCM リリースの名前。指定しない場合、「-r」オプションで指定したリリースが使用されます。

-cd <コンフィグレーション名>
ショートカットを作成するライブラリが使用している格納先 SCM 作業コンフィグレーションの名前。指定しない場合、「-c」オプションで指定したコンフィグレーションが使用されます。

-id <アイテムパス>
ショートカットを作成する格納先ライブラリのフォルダーのパス名。指定しない場合、ショートカットは格納先ライブラリのルートに作成されます。

-eud <ユーザー名>
格納先リポジトリへの接続に使用するユーザーの名前。指定しない場合、送信元リポジトリに接続しているユーザーが使用されます。

-epd <パスワード>
格納先リポジトリに接続するユーザーのパスワード。

-t <タスク ID>
格納先ライブラリに新しいショートカットを登録する 1 つ以上の LCM タスクの ID を、セミコロンで区切って指定します。ID の前部にパディングされている「0」は、省略できます。

例

ライブラリ「Sample A」のドキュメント「My Document」のショートカットを、ライブラリ「Sample B」のフォルダー「My Folder」に作成します。

```
C:¥>edms shortcut -s soop1 -d Sample -p "Sample A" -i "/MyDocument" –sd
soop1 –dd SampleB –pd "Sample B" –id "/My Folder" –m "Link my document"
```

2.8.12 restoreversion – ドキュメントバージョンの復元 [DMS]

ドキュメントの以前のバージョンを復元し、最新バージョンとして設定します。

コマンド構文：

```
edms {restoreversion | rest} -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -m <コメント> -i <アイテムパス>
      -vn <バージョン> [-po <ポート> -r <リリース名>
      -c <コンフィグレーション名> -t <タスク ID>]
```

`-s <サーバ名>`
リポジトリが存在するサーバの名前。

`-d <リポジトリ名>`
ライブラリが含まれているリポジトリの名前。

`-p <プロジェクト名>`
ライブラリの名前。

`-m <コメント>`
バージョンの復元に関する短い説明。

`-i <アイテムパス>`
バージョンを復元するドキュメントのパス名。

`-vn <バージョン>`
復元するバージョンの数値。

`-po <ポート>`
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、`9500` です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

`-r <リリース名>`
ライブラリが使用している SCM リリースの名前。指定しない場合、最初の検出可能なリリースが使用されます。

`-c <コンフィグレーション名>`
ライブラリが使用している SCM 作業コンフィグレーションの名前。指定しない場合、リリースの最初の検出可能な作業コンフィグレーションが使用されます。

`-t <タスク ID>`
変更を登録する 1 つ以上の LCM タスクの ID を、セミコロンで区切って指定します。ID の前部にパディングされている「0」は、省略できます。

例

ドキュメント「My Document」のバージョン「1」を復元します。

```
C:¥>edms restoreversion -s soop1 -d Sample -p "Sample A" -i
"/MyDocument" –m "restore my old document version" –vn "1"
```

2.8.13 delete – アイテムの削除 [DMS]

ライブラリ、および必要に応じてローカルワークスペースからアイテムを削除します。削除できるアイテムは、エディション（「-c」オプション）、フォルダー、ドキュメント、またはショートカット（「-i」オプション）です。「-c」オプションと「-i」オプションの少なくともどちらかを指定しないと削除処理ができません。

削除するアイテムがエディションまたはフォルダーの場合、含まれているすべての内容が削除されます。

コマンド構文：

```
edms {delete | del} [-f] -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -m <コメント> [-w <ローカルワークスペース名>
      -po <ポート> -r <リリース名> -c <コンフィグレーション名>
      –i <アイテムパス> -t <タスク ID>]
```

-f
確認を要求せずに、アイテムを削除します。

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
ライブラリが含まれているリポジトリの名前。

-p <プロジェクト名>
削除するアイテムが含まれているライブラリの名前。

-m <コメント>
削除に関する短い説明。

-w <ローカルワークスペース名>
削除するアイテムがチェックアウトされたローカルワークスペースのパス名。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-r <リリース名>
ライブラリが使用している SCM リリースの名前。指定しない場合、最初の検出可能なリリースが使用されます。

-c <コンフィグレーション名>
削除するライブラリエディションの名前、またはライブラリが使用している SCM コンフィグレーションの名前。作業コンフィグレーションは削除できません。

-i <アイテムパス>
削除するアイテムのパス名。指定しない場合、「–p」オプション、「-r」オプション、「–c」オプションで指定したライブラリエディションが削除されます。

-t <タスク ID>
削除を登録する 1 つ以上の LCM タスクの ID を、セミコロンで区切って指定します。ID の前部にパディングされている「0」は、省略できます。

例

ライブラリおよびローカルワークスペースから、ドキュメント「My Document」を削除します。

```
C:¥>edms delete -s soop1 -d Sample -p "Sample A" -i "/MyDocument" –m
"Delete my document" –w "C:¥myworkspace"
```

2.8.14 checkout – アイテムのチェックアウト [DMS]

ライブラリ、フォルダー、またはドキュメントをローカルワークスペースにチェックアウトします。ライブラリまたはフォルダーは ZIP ファイルとしてチェックアウトすることもできます。チェックアウトするアイテムがライブラリまたはフォルダーの場合、含まれているすべての内容がチェックアウトされます。

チェックアウトしたアイテムは、リポジトリ内で自動的にロックされます。「-ro」オプションを指定すると、アイテムは読み取り専用としてチェックアウトされ、リポジトリ内でロックされません。

コマンド構文：

```
edms {checkout | co} [-ro -f] -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> {-w <ローカルワークスペース名> |
      -z <ターゲット> [-po <ポート> -r <リリース名>
      -c <コンフィグレーション名> -i <アイテムパス>]
```

-ro
: アイテムを読み取り専用としてチェックアウトします。

-f
: 確認を要求せずに、チェックアウトした同名のアイテムにより、ローカルワークスペース内の既存のフォルダーおよびドキュメントを更新します。

-s <サーバ名>
: リポジトリが存在するサーバの名前。

-d <リポジトリ名>
: ライブラリが含まれているリポジトリの名前。

-p <プロジェクト名>
: ライブラリの名前。

-w <ローカルワークスペース名>
: アイテムをチェックアウトするローカルワークスペースのパス名。ローカルワークスペースまたは ZIP ファイルを指定してください。

-z <ターゲット>
: チェックアウトするアイテムの ZIP ファイルのパス名。ZIP ファイル名はチェックアウトされるアイテムと同じになります。

-po <ポート>
: リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-r <リリース名>
: ライブラリが使用している SCM リリースの名前。指定しない場合、最初の検出可能なリリースが使用されます。

-c <コンフィグレーション名>
: ライブラリが使用している SCM 作業コンフィグレーションの名前。指定しない場合、リリースの最初の検出可能な作業コンフィグレーションが使用されます。

-i <アイテムパス>
: チェックアウトするアイテムのパス名。指定しない場合、ライブラリのすべてのフォルダーおよびドキュメントがチェックアウトされます。

例

リポジトリからローカルワークスペースに、ドキュメント「MyDocument」を読み取り専用としてチェックアウトします。

```
C:¥>edms checkout -s soop1 -d Sample -p "Sample A" -i "/MyDocument" –w
"C:¥myworkspace" -ro
```

リポジトリからライブラリ「Sample A」を、読み取り専用の ZIP ファイルとしてチェックアウトします。

```
C:¥>edms checkout -s soop1 -d Sample -p "Sample A" -z "C:¥MyData" -ro
```

2.8.15 undocheckout – チェックアウトの取り消し [DMS]

ローカルワークスペースへのライブラリ、フォルダー、またはドキュメントのチェックアウトを取り消します。このコマンドをライブラリまたはフォルダーに適用すると、ライブラリまたはフォルダーのすべての内容が取り消されます。

チェックアウトを取り消すと、ローカルワークスペースからアイテムが削除され、リポジトリ内のアイテムのロックが解除されます。ドキュメントの場合は、最新のリポジトリバージョンが最新バージョンになります。

このコマンドは ZIP ファイルとしてチェックアウトした場合には適用できません。

コマンド構文：

```
edms {undocheckout | unco} [-f] -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -w <ローカルワークスペース名>
      [-po <ポート> -r <リリース名> -c <コンフィグレーション名>
      -i <アイテムパス> -t <タスク ID>]
```

`-f`
ローカルワークスペースでのドキュメントへのすべての変更を、確認なしで廃棄します。

`-s <サーバ名>`
リポジトリが存在するサーバの名前。

`-d <リポジトリ名>`
ライブラリが含まれているリポジトリの名前。

`-p <プロジェクト名>`
ライブラリの名前。

`-w <ローカルワークスペース名>`
アイテムがチェックアウトされているローカルワークスペースのパス名。

`-po <ポート>`
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、`9500` です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

`-r <リリース名>`
ライブラリが使用している SCM リリースの名前。指定しない場合、最初の検出可能なリリースが使用されます。

`-c <コンフィグレーション名>`
ライブラリが使用している SCM 作業コンフィグレーションの名前。指定しない場合、リリースの最初の検出可能な作業コンフィグレーションが使用されます。

`-i <アイテムパス>`
チェックアウトを取り消すアイテムのパス名。指定しない場合、このコマンドは、ライブラリのすべてのフォルダーおよびドキュメントに適用されます。

`-t <タスク ID>`
チェックアウトの取り消しを登録する 1 つ以上の LCM タスクの ID を、セミコロンで区切って指定します。ID の前部にパディングされている「0」は、省略できます。

例

ドキュメント「My Document」のチェックアウトを取り消します。

```
C:¥>edms undocheckout -s soop1 -d Sample -p "Sample A" -i "/MyDocument"
–w "C:¥myworkspace"
```

2.8.16 checkin – アイテムのチェックイン [DMS]

ドキュメントまたはフォルダーを、ローカルワークスペースからリポジトリにチェックインします。このコマンドをフォルダーに適用すると、フォルダー内のすべての内容が含まれます。

ZIP ファイルをチェックインした場合、リポジトリ内でアイテムが自動的に解凍されます。ZIP ファイル内のフォルダー構造は、チェックイン先リポジトリ内の構造と一致している必要があります。ZIP ファイル内のルートフォルダーは、チェックイン先フォルダーの名前と一致している必要があります。一致しない場合、チェックインは行われません。

命名の制限

各種のクライアントオペレーティングシステムを適正にサポートするために、フォルダーやドキュメントの名前にはすべての制御文字および以下の文字を使用できません。
`* : ¥ / ? < > | "`
フォルダーやドキュメント名にこれらの文字が含まれていると、チェックインに失敗します。

コマンド構文：

```
edms {checkin | ci} [-a -del] -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -m <コメント>
      {-w <ローカルワークスペース名> | -z <ソース>}
      [-po <ポート> -r <リリース名> -c <コンフィグレーション名>
      -i <アイテムパス> -t <タスク ID>]
```

`-a`
: リポジトリに存在しないフォルダーおよびドキュメントを、新しく追加します。このオプションを指定しないと、リポジトリに存在しないアイテムはチェックインされません。

`-del`
: ローカルワークスペースまたは ZIP ファイルに存在しないフォルダー、ドキュメントを削除します。このオプションを指定しないと、ローカルワークスペースまたは ZIP ファイルに存在しないフォルダー、ドキュメントはチェックインされません。

`-s <サーバ名>`
: リポジトリが存在するサーバの名前。

`-d <リポジトリ名>`
: ライブラリが含まれているリポジトリの名前。

`-p <プロジェクト名>`
: ライブラリの名前。

`-m <コメント>`
: チェックインに関する短い説明。

`-w <ローカルワークスペース名>`
: チェックインするアイテムまたは ZIP ファイルが存在するローカルワークスペースのパス名。ローカルワークスペースまたは ZIP ファイルを指定してください。

`-z <ソース>`

チェックインするアイテムに対する ZIP ファイル名。「–z」オプションと「–i」オプションを指定した場合、以下となります。

「–i」オプションでフォルダーを指定した場合、ZIP ファイルに含まれるすべてのアイテムがチェックインされます。

「–i」オプションでファイルを指定した場合、そのファイルを含む ZIP ファイルがリポジトリの新しいバージョンとして、チェックインされます。

`-po <ポート>`
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、`9500` です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

`-r <リリース名>`
ライブラリが使用している SCM リリースの名前。指定しない場合、最初の検出可能なリリースが使用されます。

`-c <コンフィグレーション名>`
ライブラリが使用している SCM 作業コンフィグレーションの名前。指定しない場合、リリースの最初の検出可能な作業コンフィグレーションが使用されます。

`-i <アイテムパス>`
ライブラリにチェックインするアイテムのパス名。指定しない場合、このコマンドは、ライブラリのすべてのフォルダーおよびドキュメントに適用されます。

`-t <タスク ID>`
チェックインを登録する 1 つ以上の LCM タスクの ID を、セミコロンで区切って指定します。ID の前部にパディングされている「0」は、省略できます。

例

ドキュメント「My Document」を、ローカルワークスペースからリポジトリにチェックインします。

```
C:¥>edms checkin -s soop1 -d Sample -p "Sample A" -i "/MyDocument" –w
"C:¥myworkspace"
```

2.8.17 lock – フォルダーまたはドキュメントのロック [DMS]

リポジトリ内でフォルダーまたはドキュメントをロックします。このコマンドをフォルダーに適用すると、フォルダー内のすべての内容が含まれます。

コマンド構文：

```
edms {lock | lo} -s <サーバ名> -d <リポジトリ名> -p <プロジェクト名>
      [-po <ポート> -r <リリース名> -c <コンフィグレーション名>
      -i <アイテムパス>]
```

`-s <サーバ名>`
リポジトリが存在するサーバの名前。

`-d <リポジトリ名>`
ライブラリが含まれているリポジトリの名前。

`-p <プロジェクト名>`
ライブラリの名前。

`-po <ポート>`
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、

9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-r <リリース名>
ライブラリが使用している SCM リリースの名前。指定しない場合、最初の検出可能なリリースが使用されます。

-c <コンフィグレーション名>
ライブラリが使用している SCM 作業コンフィグレーションの名前。指定しない場合、リリースの最初の検出可能な作業コンフィグレーションが使用されます。

-i <アイテムパス>
ロックするアイテムのパス名。指定しない場合、このコマンドは、ライブラリのすべてのフォルダーおよびドキュメントに適用されます。

例

リポジトリ内でドキュメント「MyDocument」をロックします。

```
C:¥>edms lock -s soop1 -d Sample -p "Sample A" -i "/MyDocument"
```

2.8.18 unlock – フォルダーまたはドキュメントのロック解除 [DMS]

リポジトリ内のフォルダーまたはドキュメントのロックを解除します。このコマンドをフォルダーに適用すると、フォルダー内のすべての内容が含まれます。

コマンド構文：

```
edms {unlock | unlo} -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> [-po <ポート> -r <リリース名>
      -c <コンフィグレーション名> -i <アイテムパス>]
```

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
ライブラリが含まれているリポジトリの名前。

-p <プロジェクト名>
ライブラリの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-r <リリース名>
ライブラリが使用している SCM リリースの名前。指定しない場合、最初の検出可能なリリースが使用されます。

-c <コンフィグレーション名>
ライブラリが使用している SCM 作業コンフィグレーションの名前。指定しない場合、リリースの最初の検出可能な作業コンフィグレーションが使用されます。

-i <アイテムパス>
ロックを解除するアイテムのパス名。指定しない場合、このコマンドは、ライブラリのすべてのフォルダーおよびドキュメントに適用されます。

例

リポジトリ内のドキュメント「MyDocument」のロックを解除します。

```
C:¥>edms unlock -s soop1 -d Sample -p "Sample A" -i "/MyDocument"
```

2.8.19 rename – アイテムの名前変更 [DMS]

リポジトリ内、および必要に応じてローカルワークスペース内のアイテムの名前を変更します。名前を変更できるアイテムは、ライブラリ（「-p」オプション、「-r」オプション、「-c」オプション）、エディション(「-c」オプション)、フォルダー、ドキュメント、またはショートカット（「-i」オプション）です。

変更管理が有効で、DMS においても設定されている場合、LCM 内の対応するアイテムの名前も変更されるか、LCM のリンクが更新されます。

命名の制限

各種のクライアントオペレーティングシステムを適正にサポートするために、アイテムの名前にはすべての制御文字および以下の文字を使用できません。
* : ¥ / ? < > | "
アイテム名にこれらの文字が含まれていると、アイテムの名前変更に失敗します。

コマンド構文：

```
edms {rename | ren} [-nolcm] -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -m <コメント> [-w <ローカルワークスペース名>
      -po <ポート> -r <リリース名> -c <コンフィグレーション名>
      -i <アイテムパス> -t <タスク ID>] <新しい名前>
```

-nolcm
: LCM のリリース管理コンポーネントへの接続を無効にします。この場合、LCM の関連するアイテムの名前は変更されず、リンクも更新されません。「変更管理への自動反映」がリポジトリで有効になっている場合は（chmansett コマンドを参照）、このオプションで接続を無効にできません。

-s <サーバ名>
: リポジトリが存在するサーバの名前。

-d <リポジトリ名>
: ライブラリが含まれているリポジトリの名前。

-p <プロジェクト名>
: ライブラリの名前。

-m <コメント>
: 名前の変更に関する短い説明。

-w <ローカルワークスペース名>
: 名前変更の対象となるローカルワークスペースのパス名。

-po <ポート>
: リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-r <リリース名>
: ライブラリが使用している SCM リリースの名前。指定しない場合、最初の検出可能なリリースが使用されます。

-c <コンフィグレーション名>
: 名前を変更するライブラリエディションの名前、またはライブラリが使用している SCM 作業コンフィグレーションの名前。指定しない場合、リリースの最初の検出可能な作業コンフィグレーションが使用されます。

-i <アイテムパス>
名前を変更するアイテムのパス名。指定しない場合、「-p」オプション、「-r」オプション、「-c」オプションで指定したライブラリまたはエディションの名前が変更されます。

-t <タスク ID>
名前変更を登録する 1 つ以上の LCM タスクの ID を、セミコロンで区切って指定します。ID の前部にパディングされている「0」は、省略できます。

<新しい名前>
変更後のアイテムの名前。

例

ドキュメント「My Document」の名前を「My Document 2」に変更します。

```
C:¥>edms rename -s soop1 -d Sample -p "Sample A" -i "/MyDocument"
"MyDocument2"
```

2.8.20 import – フォルダー、ドキュメントまたはZIPファイルのインポート [DMS]

ドキュメント、フォルダーまたは ZIP ファイルとその下位構造を、ローカルファイルシステムからリポジトリにインポートします。

インポートの対象になるのは、現在チェックアウトされているドキュメントまたはフォルダーです。必要に応じて、ローカルワークスペースを更新できます（「-w」オプション）。

コマンド構文：

```
edms {import | imp} [-f -z] -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -m <コメント> [-w <ローカルワークスペース名>
      -po <ポート> -r <リリース名> -c <コンフィグレーション名>
      -i <アイテムパス> -t <タスク ID>] <ソース>
```

-f
インポートする同名のアイテムにより、確認なしで、ライブラリにある既存のアイテムを上書きします。

-z
<送信元フォルダー名>で指定した ZIP アーカイブファイルを「-i」オプションで指定したアイテムパスに解凍してインポートします。対応する木構造がリポジトリに作成されます。指定しない場合、ZIP アーカイブファイルは解凍されることなく、まとまったファイルとしてインポートされます。

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
ライブラリが含まれているリポジトリの名前。

-p <プロジェクト名>
ライブラリの名前。

-m <コメント>
インポートに関する短い説明。

-w <ローカルワークスペース名>
処理するローカルワークスペースのパス名。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-r <リリース名>
ライブラリが使用している SCM リリースの名前。指定しない場合、最初の検出可能なリリースが使用されます。

-c <コンフィグレーション名>
ライブラリが使用している SCM 作業コンフィグレーションの名前。指定しない場合、リリースの最初の検出可能な作業コンフィグレーションが使用されます。

-i <アイテムパス>
ドキュメント、フォルダーまたは ZIP ファイルをインポートするライブラリ内のフォルダーのパス名。指定しない場合、アイテムはライブラリのルートにインポートされます。

-t <タスク ID>
インポート操作を登録する 1 つ以上の LCM タスクの ID を、セミコロンで区切って指定します。ID の前部にパディングされている「0」は、省略できます。

<ソース>
ローカルファイルシステムからインポートするドキュメント、フォルダーまたは ZIP ファイルのパス名。「-z」オプションを指定せずに ZIP ファイルをインポートした場合、１つのファイルとしての ZIP ファイルがインポートされます。

例

フォルダー「c:¥tmp」をリポジトリ内のフォルダー「My Folder」にインポートします。

```
C:¥>edms import -s soop1 -d Sample -p "Sample A" -i "/My Folder"
"C:¥tmp"
```

ZIP ファイル「c:¥MyData¥doc.zip」をリポジトリ内のフォルダー「My Folder」にインポートします。

```
C:¥>edms import .z -s soop1 -d Sample -p "Sample A" -i "/My Folder"
"C:¥MaData¥doc.zip"
```

2.8.21 export – フォルダーまたはドキュメントのエクスポート [DMS]

ドキュメントまたはフォルダーとその下位構造を、リポジトリからローカルファイルシステムまたは ZIP ファイルにエクスポートします。

エクスポートの対象になるのは、ローカルワークスペース（「-w」オプション）に現在チェックアウトされているドキュメントまたはフォルダーです。

コマンド構文：

```
edms {export | exp} [-f -z] -s <サーバ名> -d <リポジトリ名>
      -p <プロジェクト名> -i <アイテムパス> [-w <ローカルワークスペース名>
      -po <ポート> -r <リリース名> -c <コンフィグレーション名>]
      <ターゲット>
```

-f
エクスポートする同名のアイテムにより、確認なしで、格納先フォルダーにある既存のフォルダーとファイルを上書きします。

-z
「-i」オプションで指定されるアイテムを、<送信元フォルダー名>で指定される

ZIP アーカイブファイルにエクスポートします。ZIP アーカイブファイルのファイル名はエクスポートされるアイテムから作成されます。

`-s <サーバ名>`
リポジトリが存在するサーバの名前。

`-d <リポジトリ名>`
ライブラリが含まれているリポジトリの名前。

`-p <プロジェクト名>`
ライブラリの名前。

`-i <アイテムパス>`
エクスポートするドキュメントまたはフォルダーのパス名。

`-w <ローカルワークスペース名>`
処理するローカルワークスペースのパス名。

`-po <ポート>`
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、`9500` です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

`-r <リリース名>`
ライブラリが使用している SCM リリースの名前。指定しない場合、最初の検出可能なリリースが使用されます。

`-c <コンフィグレーション名>`
ライブラリが使用している SCM 作業コンフィグレーションの名前。指定しない場合、リリースの最初の検出可能な作業コンフィグレーションが使用されます。

`<ターゲット>`
アイテムのエクスポート先となるローカルファイルシステムのフォルダーのパス名。
指定したフォルダーが存在しない場合、新規に作成されます。

例

ライブラリのフォルダー「My Folder」を、ファイルシステムのフォルダー「c:¥tmp」にエクスポートします。

```
C:¥>edms export -s soop1 -d Sample -p "Sample A" -i "/My Folder"
"C:¥tmp"
```

2.8.22 changereport – 変更レポートの作成 [DMS]

指定したライブラリエディションとアイテムを比較し、すべての変更点を示す変更レポートを作成します。

変更レポートは、ライブラリ（「-p」オプション、「-r」オプション、「-c」オプション）、エディション（「-p」オプション、「-r」オプション、「-c」オプション）、フォルダーまたはドキュメント（「-i」オプション）、ドキュメントバージョン（「-i」オプション、「-vn」オプション）について作成できます。

コマンド構文：

```
edms {changereport | crpt} [-od -ov –ocr -xmlonly] -s <サーバ名>
      -d <リポジトリ名> -p <プロジェクト名> -ce <比較するエディション名>
      [-po <ポート> -r <リリース名> -c <コンフィグレーション名>
      -i <アイテムパス> –vn <バージョン> -b <辞書ファイル名>
      -x <スタイルシート名>] <レポートのファイル名>
```

-od
生成するレポートに、削除済みのドキュメントバージョンを含めます。

-ov
レポートにバージョンの詳細情報を含めます。

-ocr
レポートにタスクの詳細情報を含めます。

-xmlonly
スタイルシートや翻訳ファイルを使用しないで、XML 形式のレポートを出力します。

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
ライブラリが含まれているリポジトリの名前。

-p <プロジェクト名>
ライブラリの名前。

-ce <比較するエディション名>
比較対象とするエディションの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-r <リリース名>
ライブラリが使用している SCM リリースの名前。指定しない場合、最初の検出可能なリリースが使用されます。

-c <コンフィグレーション名>
レポートを生成するエディションの名前、またはライブラリが使用している SCM 作業コンフィグレーションの名前。指定しない場合、リリースの最初の検出可能な作業コンフィグレーションが使用されます。

-i <アイテムパス>
レポートを生成するフォルダーまたはドキュメントのパス名。このオプションと「-t」オプションをどちらも指定しない場合、「-p」オプション、「-r」オプション、「-c 」オプションで指定したライブラリまたはエディションについてレポートが生成されます。

-vn <バージョン>
レポートを作成するドキュメントのバージョンの数値。指定しない場合、最新のドキュメントバージョンについてレポートが作成されます。

-b <辞書ファイル名>
レポート生成に使用する辞書ファイルのパス名。省略すると、デフォルトの DMS 変換ファイルが使用されます。

-x <スタイルシート名>
レポート生成に使用するスタイルシートのパス名。省略すると、デフォルトの DMS スタイルシートが使用されます。

<レポートのファイル名>
出力ファイルのパス名。外部スタイルシート(「-x」オプション)または「-xmlonly」オプションを使用して異なる出力形式を指定する場合を除き、出力ファイルの拡張子は.html にする必要があります。指定したファイルがすでに存在する場合、上書きされます。

例

エディション「My Edition」と比較したドキュメント「My Document」の変更レポートを、「c:¥tmp¥report.html」というファイル名で作成します。

```
C:¥>edms changereport -s soop1 -d Sample -p "Sample A" -i "/My Document"
–ce "MyEdition" "C:¥tmp¥report.html"
```

2.8.23 historyreport – ヒストリレポートの作成 [DMS]

指定したドキュメントのヒストリレポートを作成します。

コマンド構文：

```
edms {historyreport | hrpt} [-xmlonly -oh] -s <サーバ名>
      -d <リポジトリ名> -p <プロジェクト名> -i <アイテムパス>
      [-po <ポート> -r <リリース名> -c <コンフィグレーション名>
      -vn <バージョン> -g <開始日(yyyy-MM-dd)> -h
      <終了日(yyyy-MM-dd)> -b <辞書ファイル名> -x <スタイルシート名>]
      <レポートのファイル名>
```

-xmlonly
: スタイルシートや翻訳ファイルを使用しないで、XML 形式のレポートを出力します。

-oh
: 選択したドキュメントバージョンの先行バージョンの情報だけを、レポートに追加します。このオプションを指定しないと、すべてのバージョンが含まれます。

-s <サーバ名>
: リポジトリが存在するサーバの名前。

-d <リポジトリ名>
: ライブラリが含まれているリポジトリの名前。

-p <プロジェクト名>
: ライブラリの名前。

-i <アイテムパス>
: レポートを作成するドキュメントのパス名。

-po <ポート>
: リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-r <リリース名>
: ライブラリが使用している SCM リリースの名前。指定しない場合、最初の検出可能なリリースが使用されます。

-c <コンフィグレーション名>
: ライブラリが使用している SCM 作業コンフィグレーションの名前。指定しない場合、リリースの最初の検出可能な作業コンフィグレーションが使用されます。

-vn <バージョン>
: レポートを作成するドキュメントのバージョンの数値。指定しない場合、最新のドキュメントバージョンについてレポートが作成されます。また、このオプションを指定する場合、「-oh」オプションを指定する必要があります。

-g <開始日(yyyy-MM-dd)> / –h <終了日(yyyy-MM-dd)>
: レポートに含めるヒストリ情報の開始日と終了日を、「yyyy-MM-dd」形式で指定します。指定しない場合、現在の日付までのすべてのヒストリ情報がレポートに含まれます。

-b <辞書ファイル名>
: レポート生成に使用する辞書ファイルのパス名。省略すると、デフォルトの DMS 変換ファイルが使用されます。

-x <スタイルシート名>
レポート生成に使用するスタイルシートのパス名。省略すると、デフォルトの DMS スタイルシートが使用されます。

<レポートのファイル名>
出力ファイルのパス名。外部スタイルシート(「-x」オプション)または「-xmlonly」オプションを使用して異なる出力形式を指定する場合を除き、出力ファイルの拡張子は.html にする必要があります。指定したファイルがすでに存在する場合、上書きされます。

例

ドキュメント「My Document」のヒストリレポートを、「c:¥tmp¥report.html」というファイル名で作成します。

```
C:¥>edms historyreport -s soop1 -d Sample -p "Sample A" -i "/My
Document" "C:¥tmp¥report.html"
```

2.8.24 logreport – ライブラリまたはリポジトリのログレポートの作成 [DMS]

ファイル、フォルダー、ライブラリ、またはリポジトリの監査ログを示すログレポートを作成します。

コマンド構文：

```
edms {logreport | lrpt} [-rr -xmlonly] -s <サーバ名>
      -d <リポジトリ名> -p <プロジェクト名> [-po <ポート> -r <リリース名>
      -c <コンフィグレーション名> -i <アイテムパス>
      -g <開始日(yyyy-MM-dd)> -h <終了日(yyyy-MM-dd)>
      -fa <アクション名> -fu <ユーザー名> -fn <文字列> -fp <パス名>
      -b <辞書ファイル名> -x <スタイルシート名>] <レポートのファイル名>
```

-rr
リポジトリのログレポートを作成します。このオプションを指定しないと、「-p」オプションで指定したライブラリ、または「-i」オプションで指定したアイテムのログレポートが作成されます。「-rr」オプションと「-i」オプションをどちらも指定しない場合、「-p」オプションで指定したライブラリのレポートが作成されます。

-xmlonly
スタイルシートや翻訳ファイルを使用しないで、XML 形式のレポートを出力します。

-s <サーバ名>
リポジトリが存在するサーバの名前。

-d <リポジトリ名>
ライブラリが含まれているリポジトリの名前。

-p <プロジェクト名>
ライブラリの名前。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-r <リリース名>
ライブラリが使用している SCM リリースの名前。指定しない場合、最初の検出可能なリリースが使用されます。

-c <コンフィグレーション名>
ライブラリが使用している SCM 作業コンフィグレーションの名前。指定しない場合、リリースの最初の検出可能な作業コンフィグレーションが使用されます。

-i <アイテムパス>
レポートを生成するフォルダーまたはドキュメントのパス名。レポートは、「-p」オプション、「-r 」オプション、「-c」オプションで指定したライブラリ用に生成されます。「-i」オプションと「-rr」オプションをどちらも指定しない場合、「-p」オプションで指定したライブラリのレポートが作成されます。

-g <開始日(yyyy-MM-dd)> / -h <終了日(yyyy-MM-dd)>
レポートに含めるログ情報の開始日と終了日を、「yyyy-MM-dd」形式で指定します。指定しない場合、現在の日付までのログ情報がレポートに含まれます。

-fa <アクション名>
指定したアクションによってログエントリーをフィルタリングします。<アクション名>に、アクション名をセミコロンで区切って列挙します。アクション名は利用可能なオーソリティの名前になります。利用可能なすべてのオーソリティは auth コマンドを使って表示します。

-fu <ユーザー名>
指定したユーザーによってログエントリーをフィルタリングします。<ユーザー名>に、ユーザー名をセミコロンで区切って一覧表示します。このオプションを使用できるのは、特権ユーザーだけです。

-fn <文字列>
指定した文字列によってログエントリーをフィルタリングします。<文字列>に、文字列をセミコロンで区切って列挙します。ワイルドカード「*」（1 つ以上の文字）および「?」（1 文字）を文字列内に使用できます。

-fp <パス名>
指定したパス文字列によってログエントリーをフィルタリングします。<パス名>に、パス文字列をセミコロンで区切って列挙します。ワイルドカード「*」（1 つ以上の文字）および「?」（1 文字）を文字列内に使用できます。パスセパレーター（/または¥）の相違は無視されます。

-b <辞書ファイル名>
レポート生成に使用する辞書ファイルのパス名。省略すると、デフォルトの DMS 変換ファイルが使用されます。

-x <スタイルシート名>
レポート生成に使用するスタイルシートのパス名。省略すると、デフォルトの DMS スタイルシートが使用されます。

<レポートのファイル名>
出力ファイルのパス名。外部スタイルシート(「-x」オプション)または「-xmlonly」オプションを使用して異なる出力形式を指定する場合を除き、出力ファイルの拡張子は.html にする必要があります。指定したファイルがすでに存在する場合、上書きされます。

例

リポジトリのログレポートを、「c:¥tmp¥report.html」というファイル名で作成します。

```
C:¥>edms logreport -s soop1 -d Sample -p "Sample A" –rr
"C:¥tmp¥report.html"
```

2.9 ライフサイクル管理

以下のセクションでは、LCM で使用できるコマンドについて説明します。これらのコマンドを呼び出すには、`elcm.bat` バッチファイルまたは `elcm.sh` シェルスクリプトを使用します。

各コマンドに、リポジトリ接続情報を指定できます。また、`setconnection` コマンドを使用して、事前にこの情報を保存しておくこともできます。詳細については、「1.3.6 LCM の接続情報の保存」を参照してください。

2.9.1 createtask – タスクの作成 [LCM]

指定したアプリケーションのタスクを作成します。タスクの名前は自動的に生成され、作成後に表示されます。

コマンド構文：

```
elcm {createtask | ct}
      -ap <アプリケーション名>
      [-pd <プロセス名> -tt <タスクタイプ>]
```

-ap <アプリケーション名>
　タスクを作成するアプリケーションの名前。

-pd <プロセス名>
　新しいタスクに割り当てるプロセスの名前。プロセス名は showprocesses コマンドで表示される形式で指定します。指定しない場合、あらかじめ定義されているアプリケーションのデフォルトプロセスが割り当てられます。

-tt <タスクタイプ>
　使用するタスクタイプの名前。あらかじめ定義されているモデル定義には、「Task」、「BugfixTask」および「EnhancementTask」が用意されています。指定しない場合、「Task」タスクタイプが割り当てられます。
　モデル定義ファイルにあるタスクタイプの名前を指定します。モデル定義の詳細およびカスタマイズ方法の詳細については、「カスタマイズガイド」を参照してください。

例

デフォルトプロセスを使用して、アプリケーション「MyProduct」のタスクを作成します。

```
C:¥>elcm ct –p MyProduct
```

プロセス「myTaskProcess」を使用して、アプリケーション「MyProduct」のタスクを作成します。

```
C:¥>elcm ct –p MyProduct –pd myTaskProcess
```

2.9.2 listitems – タスクの一覧表示 [LCM]

アプリケーションに含まれる指定したタスクを、プロパティを含めて一覧表示します。各種フィルターを定義して、各タスクのプロパティを表示できます。

コマンド構文：

```
elcm { listitems | li}
      -ap <アプリケーション名> –t
      [-nf <フィルター> -pn <プロパティ名>] [-lru] [-lst] [–latt]
      [–lpi]
```

-ap <アプリケーション名>
タスクが割り当てられているアプリケーションの名前。

-t
タスクのリストを一覧表示します。

-nf <フィルター>
一覧表示されるタスクの ID によるフィルター。ID を指定するか、ワイルドカードを指定して ID のパターンを指定します。ワイルドカード「*」は任意の数の文字、「?」は任意の 1 文字として代用できます。

-pn <プロパティ名>
タスクについて表示するプロパティを、セミコロンで区切って列挙します。このオプションに「all」を指定するか、オプション全体を省略した場合、すべてのプロパティが表示されます。プロパティ名は、モデル定義ファイルで指定されています。このファイルの「attributes」セクションにおいて、それぞれの LCM のアイテムごとに、対応するプロパティが定義されています。モデル定義、およびそのカスタマイズ方法の詳細については、「カスタマイズガイド」を参照してください。

-lru
各タスクの責任元ユーザーおよびロールを一覧表示します。

-lst
各タスクのステータスおよび個々のステータスの責任元ユーザーを一覧表示します。

-latt
各タスクの添付ファイルを一覧表示します。添付ファイルはタイトルとファイル名が一覧表示されます。

-lpi
各タスクのプロセス定義名および現在のステートを表示します。

例

アプリケーション「MyApplicatoin」のすべてのタスクと添付ファイルを一覧表示します。

```
C:¥> elcm li –ap MyApplication –t –latt



プロパティ:
ID: 0004711
概要: 機能 x の実装
タイプ: タスク
アプリケーション: MyApplication
親タスク:
責任元ユーザー: Administrator
優先度: P1 最高
期限: 2009/12/26
予定工数: 2 週
作成者: Administrator
作成日時: 2009/11/26 10:26:20
修正者: Administrator
修正日時: 2009/11/26 10:29:54
説明:
```

```
コメント:


添付ファイル:
タスク計画 (ファイル名: myTaskPlan.xls)
タスク詳細 (ファイル名: myTask.doc)
```

2.9.3 getitem – タスクのプロパティの一覧表示 [LCM]

指定したタスクのプロパティを一覧表示します。

コマンド構文：

```
elcm {getitem | gi} -t <タスク ID>
      [-pn <プロパティ名>] [-lru] [-lst] [-latt] [-lpi]
      [-lci <フィルター>]
```

`-t <タスク ID >`
プロパティを一覧表示するタスクの ID。ID の前部にパディングされている「0」は、省略できます。

`-pn <property>`
指定したタスクについて表示するプロパティを、セミコロンで区切って列挙します。このオプションに「all」を指定するか、オプション全体を省略した場合、すべてのプロパティが表示されます。プロパティ名は、モデル定義ファイルで指定されています。このファイルの「attributes」セクションにおいて、それぞれの LCM のアイテムに、対応するプロパティが定義されています。モデル定義およびそのカスタマイズ方法の詳細については、「カスタマイズガイド」を参照してください。

`-lru`
指定したタスクの責任元ユーザーおよびロールを一覧表示します。

`-lst`
指定したタスクのステータスおよび個々のステータスの責任元ユーザーを一覧表示します。

`-latt`
指定したタスクの添付ファイルを一覧表示します。添付ファイルはタイトルとファイル名が一覧表示されます。

`-lpi`
指定したタスクのプロセス定義名、および現在のステートを表示します。

`-lci`
タスクに割り当てられている変更アイテムを表示します。名前でフィルターをかける事ができます。フィルターを使用しない場合、すべての変更アイテムが一覧表示されます。

`<フィルター>`
変更アイテムの名前によるフィルター。名前を指定するか、ワイルドカードを指定して名前のパターンを指定します。ワイルドカード「*」は任意の数の文字、「?」は任意の 1 文字として代用できます。

例

タスク「0004711」のプロパティを一覧表示します。

```
C:¥>elcm gi –t 0004711

プロパティ:
ID: 0004711
```

```
概要: 機能 x の実装
タイプ: タスク
アプリケーション: MyApplication
親タスク: 0004700
責任元ユーザー: Administrator
優先度: P1
最高期限: 2009/12/26
予定工数: 2 週
作成者: Administrator
作成日時: 2009/11/26 10:26:20
修正者: Administrator
修正日時: 2009/11/26 10:29:54
説明:
カテゴリー:
コメント:
```

2.9.4 setproperties – タスクのプロパティの更新 [LCM]

特定のタスクのプロパティを、現在のステート、または特定のステートで更新します。

コマンド構文：

```
elcm {setproperties | sp} –t <タスク名> [-st <ステート名>]
      [-u <ユーザー名> [-du]] [-r <ロール名> [-ru]]
      [-pn <プロパティ名> -pv <プロパティ値>]
```

-t <タスク名>
: プロパティを変更するタスクの ID。先頭にゼロを付けないで ID を入力します。

-st <ステート名>
: 責任元ユーザーとロールを変更するステートの名前。ステート名は showprocesses コマンドで表示される形式で指定します。指定しない場合、責任元ユーザーとロールがタスクの現在のステートに対して変更されます。

-u <ユーザー名>
: 指定したステートの責任元ユーザーとして割り当てるユーザーを、セミコロンで区切って列挙します。

-du
: 指定したステートから、「– u」オプションで指定した責任元ユーザーの割り当てを解除します。

-r <ロール名>
: 指定したステートの責任元ロールとして設定するロールを、セミコロンで区切って列挙します。

-dr
: 指定したステートから、「– r」オプションで指定した責任元ロールの割り当てを解除します。

-pn <プロパティ名>
: 値を変更するプロパティを、セミコロンで区切って列挙します。プロパティ名は getitem コマンドで表示される形式で指定します。

-pv <プロパティ値>
: 「– pn」オプションで指定したプロパティの、新しいプロパティ値を、セミコロンで区切って列挙します。プロパティとそれに対応する値は、リスト内で位置を一致させます。

注意：セミコロン（;）を含む値を指定する場合は、「-pv "a¥¥;b"」のように、セミコロンの前に ¥ を 2 つ入力します。

データ型によって、以下の値を指定できます。

- 文字列型：任意の文字列
- 論理型：論理値（`true` または `false`）
- 日付型：`yyyy-MM-dd` 形式の日付
- 期間型：ISO 8601 に準拠した期間の値。<n>年に対して `P<n>Y` を、<n>月に対して `P<n>M` を、<n>日に対して `P<n>D` を、または<n>時間に対して `PT<n>H` を指定できます。例えば、`P17D` は 17 日の期間を定義します。週、分、および秒の指定は未サポートです。また、年、月、日、および時間の組み合わせも未サポートです。
- 付加文字列型：現在の値に付加する任意の文字列。
- 整数型：整数
- 浮動型：浮動小数点数
- カテゴリー：LCM で定義されているカテゴリーのカンマ区切りリスト

例

タスク「0004711」のステート「Implementation」に責任元ユーザーを設定します。

```
C:¥>elcm sp –t 4711 –ac Implementation –u server1¥Smith;server1¥Miller
```

タスク「0004711」の「Short Description」、「Description」、「Priority」の各プロパティに新しい値を設定します。

```
C:¥>elcm sp –t 4711 –pn "shortDescription;Description;priority" – pv "my
new short description;my new long description;P2"
```

2.9.5 changestate – タスクのステートの変更 [LCM]

指定したタスクの、現在のステートを変更します。または、プロセスを再開します。

コマンド構文：

```
elcm {changestate | cs} – t <タスク名> [-st <ステート名> | -pd
      <プロセス名>] [-u <ユーザー名>]
```

-t <タスク名>
: ステートを変更するタスクの ID。先頭にゼロを付けないで ID を入力します。

-st <ステート名>
: タスクに割り当てるステートの名前。ステート名は `showprocesses` コマンドで表示される形式で指定します。

-pd <プロセス名>
: 再開するプロセスの名前。プロセス名は `showprocesses` コマンドで表示される形式で指定します。

-u <ユーザー名>
: 「– st」オプションで指定したステートの責任元ユーザーとして設定するユーザーを、セミコロンで区切って列挙します。指定しない場合、かつ、現在のステートが、再開したプロセスの一部である場合は、最初に割り当てたユーザーが責任元ユーザーとして設定されます。そうでない場合は、ステートのデフォルトロールが割り当てられます。その結果、このロールを持つすべてのユーザーが責任元ユーザーになります。

例

タスク「0004711」の現在のステートを「Implementation」に変更します。

```
C:¥>elcm cs –t 4711 –st "Implementation"
```

2.9.6 delete – タスクの削除 [LCM]

指定したタスクを削除します。

コマンド構文：

```
elcm {delete | del} – t <タスク ID>
```

-t <タスク ID>
削除するタスクの ID。ID の前部にパディングされている「0」は、省略できます。

例

タスク「0004711」を削除します。

```
C:¥>elcm del –t 4711
```

2.9.7 assignrelease – リリースへのタスクの割当て [LCM]

指定したタスクを特定のリリースに割り当てます。タスクとリリースは、同じアプリケーションに属している必要があります。

コマンド構文：

```
elcm {assignrelease | arl} –t <タスク ID> -rl <リリース名>
```

-t <タスク ID>
「–rl」オプションで指定したリリースに割り当てるタスクの ID。ID の前部にパディングされている「0」は、省略できます。

-rl <リリース名>
「–t」オプションで指定したタスクを割り当てるリリースの名前。

例

タスク「0004711」をリリース「RL1」に割り当てます。

```
C:¥>elcm arl –t 4711 –rl RL1
```

2.9.8 importattachment – 添付ファイルのインポート [LCM]

ファイルシステムから添付ファイルをタスクにインポートします。

コマンド構文：

```
elcm {importattachment | iat} -t <タスク ID> –f <ファイル名>
      -att <添付ファイル名>
```

-t <タスク ID>
添付ファイルをインポートするタスクの ID。ID の前部にパディングされている「0」は、省略できます。

-f <ファイル名>
タスクに、添付ファイルとしてインポートするファイルの絶対パス名。

-att <添付ファイル名>
リポジトリ内の添付ファイルのタイトル。

例

タスク「0004711」に添付ファイル「project plan」をインポートします。

```
C:¥>elcm iat -t 4711 -f "c:¥attachments¥projectplan.xls" -att "project
plan"
```

2.9.9 exportattachment – 添付ファイルのエクスポート [LCM]

タスクから、指定した添付ファイルをファイルシステム内の指定された格納先にエクスポートします。

コマンド構文：

```
elcm {exportattachment | eat} -t <タスク ID>
      -att <添付ファイル名> –f <ファイル名>
```

-t <タスク ID>
　添付ファイルをエクスポートするタスクの ID。ID の前部にパディングされている「0」は、省略できます。

-att <添付ファイル名>
　リポジトリ内の添付ファイルのタイトル。

-f <ファイル名>
　添付ファイルのエクスポート先となるフォルダーまたはファイルの絶対パス名。指定したフォルダー構造はすでにファイルシステム上に存在する必要があります。すでに存在するファイルを指定した場合、ファイルは上書きされます。

例

タスク「0004711」の添付ファイル「project plan」を、「c:¥attachments¥」フォルダーにエクスポートします。

```
C:¥>elcm eat -t 4711 -att "project plan" -f "c:¥attachments"
```

2.9.10 deleteattachment – 添付ファイルの削除 [LCM]

タスクから、指定した添付ファイルを削除します。

コマンド構文：

```
elcm {deleteattachment | dat} –t <タスク ID> –att <添付ファイル名>
```

-t <タスク ID>
　添付ファイルを削除するタスクの ID。ID の前部にパディングされている「0」は、省略できます。

-att <添付ファイル名>
　リポジトリ内の添付ファイルのタイトル。

例

タスク「0004711」から添付ファイル「project plan」を削除します。

```
C:¥>elcm dat -t 4711 -att "project plan"
```

2.9.11 setlinks – リンクの更新 [LCM]

タスクと変更アイテムとの関連などの ADM コンポーネントのアイテムとのリンク、または外部システムとのリンクを更新します。リンクは次のパラメーターにより指定されます。

- サーバ名

- ポート
- リポジトリ(ADM コンポーネントのアイテムとのリンクのみ)
- プロジェクト(SCM または DMS のアイテムとのリンクのみ)
- リリース (SCM または DMS のアイテムとのリンクのみ)
- コンフィグレーション(SCM または DMS のアイテムとのリンクのみ)

o で始まるコマンドパラメーター（-os, -opo, -od, -op, -or, -oc）は、変更されるリンクを取得するための検索条件を指定できます。検索されたすべてのリンクに対して、d で終わるコマンドパラメーター（-sd, -pod, -dd, -pd, -rd, -cd）で指定した値で、対応する元の値が置き換えられます。

例えば、リポジトリの位置が変更されたためリンクを更新しなければならない場合、リンクされたアイテムに含まれる元のリポジトリ名と変更後のリポジトリ名を指定します。元の値を指定する場合、ワイルドカードを使って指定することができます。元の値、変更後の値は少なくとも 1 つ定義する必要があります。

注意：この操作を実行するユーザーは、リポジトリに対する Administrate オーソリティを保持していることを推奨します。Administrate 権限を保持していない場合には、影響を受けるすべてのアイテムに対する Edit オーソリティを保持しているかチェックが行われるため、パフォーマンスの低下を招きます。

コマンド構文：

```
elcm {setlinks | sl} [-os <サーバ名> -opo <ポート> -od <リポジトリ名>
      -op <プロジェクト名> -or <リリース名> -oc <コンフィグレーション名>
      -sd <サーバ名> -pod <ポート> -dd <リポジトリ名>
      -pd <プロジェクト名> -rd <リリース名> -cd <コンフィグレーション名>
      -test]
```

-os <サーバ名>
リンクに含まれる更新前のサーバの名前。

-opo <ポート>
リンクに含まれる更新前のサーバのポート。

-od <リポジトリ名>
リンクに含まれる更新前のリポジトリの名前。このオプションには常にリポジトリ管理名を指定する必要があります。

-op <プロジェクト名>
リンクに含まれる更新前の SCM プロジェクト、または DMS ライブラリの名前。

-or <リリース名>
リンクに含まれる更新前の SCM リリースの名前。

-oc <コンフィグレーション名>
リンクに含まれる更新前の SCM コンフィグレーションの名前。

-sd <サーバ名>
リンクに設定される更新後のサーバの名前。

-pod <ポート>
リンクに設定される更新後のポート。

-dd <リポジトリ名>
リンクに設定される更新後のリポジトリの名前。このオプションには常にリポジトリ管理名を指定する必要があります。

-pd <プロジェクト名>
リンクに設定される更新後の SCM プロジェクト、または DMS ライブラリの名前。

-rd <リリース名>
リンクに設定される更新後の SCM リリースの名前。

-cd <コンフィグレーション名>
リンクに設定される更新後の SCM コンフィグレーションの名前。

-test
まだリポジトリに登録されていない新しいリンク定義を表示します。このオプションを使うことで、更新前にコマンドによる更新内容をみることができます。

例

SCM アイテムへのすべてのリンクのリリース名を「1.0」から「1.0Released」に更新します。

```
C:¥>elcm sl -or 1.0 -rd 1.0Released
```

サーバ名が「server1」、ポートが「9501」、リポジトリが「rep1」、およびプロジェクトが「Proj1」であるリンクに対してのみ、リリース名を「1.0」から「1.0Released」に変更します。

```
C:¥>elcm sl -os server1 -op 9501 -od rep1 -op Proj1 -or 1.0 –rd
1.0Released
```

「fileserver」で始まるサーバ名を含むすべてのリンクに対して、サーバ名を「fileserver」から「bigfileserver.biz」に、ポートを「9700」に変更します。

```
C:¥>elcm sl –os ¥"fileserver*¥" –sd bigfileserver.biz –pod 9700
```

2.9.12 summaryreport – サマリレポートの作成 [LCM]

指定したタスクのサマリレポートを作成します。

コマンド構文：

```
elcm {summaryreport | sr} -t <タスク ID>
      -f <ファイル名> [-of <ファイル形式>]
```

-t <タスク ID>
サマリレポートを作成するタスクの ID。ID の前部にパディングされている「0」は、省略できます。

-f <ファイル名>
サマリレポートの出力ファイルの絶対パス名。指定したファイルがすでに存在する場合、上書きされます。

-of <ファイル形式>
サマリレポートのファイル形式。使用できるファイル形式は、HTML または XML です。デフォルトのファイル形式は、HTML です。

例

タスク「0004711」のサマリレポートを、ファイル「c:¥reports¥summaryreport.html」へ出力します。

```
C:¥>elcm sr -t 4711 -f "c:¥reports¥summaryreport.html"
```

2.9.13 logreport – ログレポートの作成 [LCM]

リポジトリ、アプリケーション、またはタスクの監査ログを作成します。

コマンド構文：

```
elcm {logreport | lr} [-ap <アプリケーション名> | -t <タスク ID>] -f <
      ファイル名>
      [-of <ファイル形式> -fr <開始日> -to <終了日> -u <ユーザー名>]
```

-ap <アプリケーション名>
ログレポートを作成するアプリケーションの名前。-p も-t も指定しない場合、リポジトリ全体のログレポートが作成されます。

-t <タスク ID>
ログレポートを作成するタスクの ID。ID の前部にパディングされている「0」は、省略できます。

-f <ファイル名>
ログレポートの出力先ファイルの絶対パス名。指定したファイルがすでに存在する場合、上書きされます。

-of <ファイル形式>
ログレポートのファイル形式。使用できるファイル形式は、HTML または XML です。デフォルトのファイル形式は、HTML です。

-fr <開始日>
ログに出力する期間の開始日を、yyyy-MM-dd 形式で指定します。

-to <終了日>
ログに出力する期間の終了日を、yyyy-MM-dd 形式で指定します。

-u <ユーザー名>
ログにアクションを出力するユーザーの名前をフィルタリングできます。ユーザー名をセミコロンで区切って指定します。

例

タスク「0004711」のログレポートを、ファイル「c:¥reports¥mytask_logreport.html」へ出力します。

```
C:¥>elcm lr –t 4711 –f "c:¥reports¥mytask_logreport.html"
```

2.9.14 importprocesses – プロセス定義のインポート [LCM]

XML ファイルから LCM リポジトリにプロセス定義をインポートします。

デフォルトプロセスは、プロセス定義ファイルで指定されています。プロセス定義、およびそのカスタマイズ方法の詳細については、「カスタマイズガイド」を参照してください。

コマンド構文：

```
elcm {importprocesses | iproc} -f <ファイル名>
```

-f <ファイル名>
プロセス定義を含む XML ファイルの絶対パス名。

例

「c:¥process¥processimport.xml」ファイルからプロセス定義をインポートします。

```
C:¥>elcm iproc –f "c:¥process¥processimport.xml"
```

2.9.15 exportprocesses – プロセス定義のエクスポート [LCM]

LCM リポジトリから XML ファイルにすべてのプロセス定義をエクスポートします。このファイルは他の LCM リポジトリにインポートするか、または新規 LCM リポジトリのテンプレートとして使用できます。

デフォルトプロセスは、プロセス定義ファイルで指定されています。プロセス定義およびそのカスタマイズ方法の詳細については、「カスタマイズガイド」を参照してください。

コマンド構文：

```
elcm {exportprocesses | eproc} -f <ファイル名>
```

-f <ファイル名>
プロセス定義のエクスポート先となる XML ファイルの絶対パス名。指定したファイルがすでに存在する場合、上書きされます。

例

プロセス定義を「c:¥process¥processexport.xml」ファイルにエクスポートします。

```
C:¥>elcm eproc –f "c:¥process¥processexport.xml"
```

2.9.16 showprocesses – プロセス定義の表示 [LCM]

リポジトリ内で使用できるプロセス定義のリストをステートを含めて表示します。

デフォルトプロセスは、プロセス定義ファイルで指定されています。プロセス定義およびそのカスタマイズ方法の詳細については、「カスタマイズガイド」を参照してください。

コマンド構文：

```
elcm {showprocesses | sproc}
```

例

現在のリポジトリのプロセス定義を一覧表示します。

```
C:¥>elcm sproc

Available process definitions:
Iteration: Standard process
       Planning -> Implementation, Canceled
       Implementation -> Planning, Closed
       Closed
       Canceled

Product: Standard process
       Development -> Closed
       Closed

Release: Standard process
       Planning -> Implementation, Canceled
       Implementation -> Closed, Planning
       Closed
       Canceled

Requirement: Standard process
       Definition -> Rejected, Review
       Review -> Definition, Rejected, Planning
       Planning -> Implementation, Rejected, Definition
```

```
       Implementation -> Planning, Complete, Rejected
       Rejected
       Complete

Task: Bugzilla process
       Assigned -> Moved, Fixed, Works for me, Duplicate, Invalid, Fix
later,

Won't fix, Remind
       Fixed
       Invalid
       Won't fix
       Fix later
       Remind
       Duplicate
       Works for me
       Moved

Task: Standard process
       Definition -> Rejected, Implementation
       Implementation -> Complete, Definition
       Complete
       Rejected
```

2.10 検索インデックス

以下のコマンドは、LCM の検索とクエリ、および SCM と DMS の全文検索用の検索インデックスの作成および保守に使用します。

これらのコマンドを呼び出すには、`esindex.bat` バッチファイルを使用します。

注意：ADM クライアントインストーラを使用してコマンドラインインターフェースをインストールした場合、`esindex` コマンドはインストールされません。`esindex` コマンドは検索サーバ上で直接実行する必要があるためです。このコマンドをインストールするには、サーバインストーラを使用してください。検索サーバと ADM のインデックスについての詳細は「管理者ガイド」を参照してください。

2.10.1 create – 検索インデックスの作成 [SCM、DMS]

新しい検索インデックスを作成します。以下の検索インデックスを作成できます。

- SCM または DMS リポジトリ全体
- 個々の SCM プロジェクト
- 個々の DMS ライブラリ
- インデックス作成済みの SCM プロジェクトおよび DMS ライブラリ内の、個々のフォルダーとファイル（ドキュメント）

コマンド構文：

```
esindex create -s <サーバ名> -d <リポジトリ名> -eu <ユーザー名>
      -ep <パスワード> [-po <ポート> -p <プロジェクト名>
      -i <アイテムパス> -t <インデックスパス> -l <ログファイル名>]
```

`-s <サーバ名>`
検索インデックスを作成するリポジトリがあるサーバの名前。

`-d <リポジトリ名>`
検索インデックスを作成するリポジトリのリポジトリ管理名。

`-eu <ユーザー名>`
リポジトリに接続するユーザーの名前。このユーザーは、リポジトリ内のインデックスに含まれるすべてのデータについて、読み取り権限を持っている必要があります。

`-ep <パスワード>`
ユーザーのパスワード。

`-po <ポート>`
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、`9500` です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

`-p <プロジェクト名>`
SCM プロジェクトまたは DMS ライブラリの名前。指定したプロジェクトまたはライブラリだけに検索インデックスが作成されます。指定しない場合、リポジトリ内のすべてのプロジェクトやライブラリの検索インデックスが対象になります。

`-i <アイテムパス>`
プロジェクトまたはライブラリのファイル（ドキュメント）またはフォルダーのパス名。パス区切り文字にはスラッシュ（/）を使用してください。指定したファイルのすべてのバージョン、または指定したフォルダーのすべてのファイルバージョン

に対してのみ、検索インデックスが作成されます。「-p」オプションで、関連するプロジェクトまたはライブラリを指定する必要があります。

-t <インデックスパス>
検索サーバ上の検索インデックスを保存するフォルダーの絶対パス名。ADM コンポーネントにより、以下のフォルダーとリポジトリ名を使用することが推奨されます：インストール時に指定したアプリケーションデータフォルダー（通常 Windows の場合 C:¥InterstageADM、UNIX/Linux の場合/var/opt/FJSViasm）のサブフォルダーsearchIndex¥SCM または searchIndex¥DMS。このオプションを省略した場合、対応するリポジトリプロパティに指定されているインデックスパスが使用されます。

-l <ログファイル名>
インデックス処理のログファイルの絶対パス名。指定しない場合、「-t」オプションで指定したフォルダー内の「index.log」ファイルにログが書き込まれます。

2.10.2 update – 検索インデックスの更新 [SCM、DMS]

索引インデックスの作成後または最終更新後、指定した SCM または DMS のリポジトリで追加変更されたデータにより、索引インデックスを更新します。

コマンド構文：

```
esindex update [-visible] -s <サーバ名> -d <リポジトリ名>
      -eu <ユーザー名> -ep <パスワード> [-po <ポート>
      -p <プロジェクト名> -t <インデックスパス> -l <ログファイル名>]
```

-visible
ファイルまたはドキュメントの表示可能バージョンの検索インデックスだけを更新します。このオプションを指定しないと、必要に応じて、すべてのバージョンの検索インデックスが更新されます。

-s <サーバ名>
検索インデックスを更新するリポジトリがあるサーバの名前。

-d <リポジトリ名>
検索インデックスを更新するリポジトリのリポジトリ管理名。

-eu <ユーザー名>
リポジトリに接続するユーザーの名前。このユーザーは、リポジトリ内のインデックスに含まれるすべてのデータについて、読み取り権限を持っている必要があります。

-ep <パスワード>
ユーザーのパスワード。

-po <ポート>
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、9500 です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

-p <プロジェクト名>
SCM プロジェクトまたは DMS ライブラリの名前。指定したプロジェクトまたはライブラリの検索インデックスだけが更新されます。指定しない場合、リポジトリ内のすべてのプロジェクトやライブラリの検索インデックスが対象になります。

-t <インデックスパス>
更新する検索インデックスが存在する、検索サーバ上のフォルダーの絶対パス名。ADM コンポーネントにより、以下のフォルダーとリポジトリ名を使用することが推奨されます：インストール時に指定したアプリケーションデータフォルダー（通常 Windows の場合 C:¥InterstageADM、UNIX/Linux の場合/var/opt/FJSViasm）のサブ

フォルダーsearchIndex¥SCM または searchIndex¥DMS。このオプションを省略した場合、対応するリポジトリプロパティに指定されているインデックスパスが使用されます。

`-l <ログファイル名>`
インデックス処理のログファイルの絶対パス名。指定しない場合、「`-t <インデックスパス>`」で指定したフォルダー内の「`index.log`」ファイルにログが書き込まれます。

2.10.3 delete – 検索インデックスのエントリーの削除 [SCM、DMS]

検索インデックスからデータを削除します。検索インデックスのエントリーを削除するには、指定した SCM または DMS のリポジトリ、プロジェクト、またはライブラリが存在している必要があります。つまり、プロジェクト、ライブラリ、またはリポジトリを削除する前に、検索インデックスを削除する必要があります。

削除したエントリーは、インデックスの最適化によってのみ、検索インデックスから物理的に削除されます。

注意：リポジトリのインデックス全体を削除したい場合は、単純に、検索サーバ上の対応するインデックスファイルまたはフォルダーを削除します。

コマンド構文：

```
esindex delete -s <サーバ名> -d <リポジトリ名> -eu <ユーザー名>
      -ep <パスワード> [-po <ポート> -p <プロジェクト名>
      -t <インデックスパス> -l <ログファイル名>]
```

`-s <サーバ名>`
検索インデックスを削除するリポジトリがあるサーバのリポジトリ管理名。

`-d <リポジトリ名>`
検索インデックスを削除するリポジトリのリポジトリ管理名。

`-eu <ユーザー名>`
リポジトリに接続するユーザーの名前。

`-ep <パスワード>`
ユーザーのパスワード。

`-po <ポート>`
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、`9500` です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

`-p <プロジェクト名>`
SCM プロジェクトまたは DMS ライブラリの名前。指定したプロジェクトまたはライブラリの検索インデックスだけが削除されます。指定しない場合、リポジトリ内のすべてのプロジェクトまたはリポジトリのインデックスが削除されます。

`-t <インデックスパス>`
検索インデックスが存在する、検索サーバ上のフォルダーの絶対パス名。このオプションを省略した場合、対応するリポジトリプロパティに指定されているインデックスパスが使用されます。

`-l <ログファイル名>`
インデックス処理のログファイルの絶対パス名。指定しない場合、「`-t`」で指定したフォルダー内の「`index.log`」ファイルにログが書き込まれます。

2.10.4 optimize – 検索インデックスの最適化 [SCM、DMS]

SCM または DMS リポジトリの検索インデックスを最適化します。このコマンドを使用することにより、内部インデックス構造が最適化され、削除したエントリーが物理的に削除されて、検索インデックスの物理サイズが減少し、検索処理の速度が向上します。

検索インデックスの作成を行うと、複数のインデックスファイルが作成されます。インデックスファイルの数が増加すると、検索の速度が落ちてしまいます。最適化することで、インデックスファイルを結合し、検索の速度を向上させます。

検索速度が落ちた場合や、インデックスの更新を行った後には、最適化を実行するようにしてください。

最適化の実行中には、ディスク入出力が頻繁に発生するため、システムをあまり使用しない時間に実行することを推奨します。インデックスの作成中や更新中には最適化を実行しないでください。インデックスが破損することはありませんが、作成や更新にかかる時間が増加します。

コマンド構文：

```
esindex optimize {-s <サーバ名> -po <ポート> -d <リポジトリ名>
      -eu <ユーザー名> -ep <パスワード> | -t <インデックスパス>}
      [-l <ログファイル名>]
```

`-s <サーバ名>`
検索インデックスを最適化するリポジトリがあるサーバの名前。「`-t`」オプションを指定する場合、このオプションを指定する必要はありません。

`-po <ポート>`
リポジトリサーバにアクセスするポート。初期状態でのデフォルトのポートは、`9500` です。デフォルトポートの変更方法の詳細については、「管理者ガイド」を参照してください。

`-d <リポジトリ名>`
検索インデックスを最適化するリポジトリのリポジトリ管理名。「`-t`」オプションを指定する場合、このオプションを指定する必要はありません。

`-eu <ユーザー名>`
リポジトリに接続するユーザーの名前。「`-t`」オプションを指定する場合、このオプションを指定する必要はありません。

`-ep <パスワード>`
ユーザーのパスワード。

`-t <インデックスパス>`
最適化する検索インデックスが存在する、検索サーバ上のフォルダーの絶対パス名。このオプションを省略した場合、対応するリポジトリプロパティに指定されているインデックスパスが使用されます。

`-l <ログファイル名>`
インデックス処理のログファイルの絶対パス名。指定しない場合、「`-t`」で指定したフォルダー内の「`index.log`」ファイルにログが書き込まれます。

2.10.5 rebuild – LCM検索インデックスの再構成 [LCM]

LCM リポジトリの検索インデックスを再構成します。特定の状況下では、LCM リポジトリのインデックスの再構成が必要な場合があります。例えば、リポジトリのデータを大量

に削除したため、関連するインデックスのエントリーも削除したい場合や、リポジトリやサーバの名前を変更する場合などです。

コマンド構文：

```
esindex rebuild -d <リポジトリ名> -ru <リモート URL> -eu <ユーザー名>
      -ep <パスワード>
```

-d <リポジトリ名>
　検索インデックスを再構成するリポジトリの名前。

-ru <リモート URL>
　リモートインターフェースの URL を、以下の形式で指定します。
　http[s]://<サーバ>[:<ポート>]
　<サーバ名>は、ilcmserver<バージョン>Web アプリケーションが配置されているアプリケーションサーバです。<ポート>は、サーバのアドレスを示すポート番号です。デフォルトでは、HTTP では 80、HTTPS では 443 が使用されます。

-eu <ユーザー名>
　リポジトリに接続するユーザーの名前。このユーザーは、リポジトリ内のインデックスに含まれるすべてのデータについて、読み取り権限を持っている必要があります。

-ep <パスワード>
　ユーザーのパスワード。